AFIT/GSM/ENC/92S-9

INTERACTIVE GRAPHICS SYSTEM FOR THE STUDY OF VARIANCE/COVARIANCE STRUCTURES OF BIVARIATE AND MULTIVARIATE NORMAL POPULATIONS

THESIS

Ronald G. Garlicki, Captain, USAF

AFIT/GSM/ENC/92S-9

The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

AFIT/GSM/ENC/92S-9

INTERACTIVE GRAPHICS SYSTEM FOR THE STUDY OF VARIANCE/COVARIANCE STRUCTURES OF BIVARIATE AND MULTIVARIATE NORMAL POPULATIONS

THESIS

Presented to the Faculty of the School of Systems and Logistics
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Systems Management

Ronald G. Garlicki, B.S.
Captain, USAF

September 1992

Acknowledgements

I would like to thank my beautiful and loving wife Diana for her support and understanding through the long days and longer nights spent at the computer while conducting this research.

I would like to thank Professor Dan Reynolds and Mr. Richard Lamb because without their ideas and assistance this research could not have been completed.

I would also like to thank Tracy Sommers for her time and speedy typing fingers and Bill for lending me Tracy, his "military spouse."

Ronald G. Garlicki

Table of Contents

List of Figures

List of Tables

AFIT/GSM/ENC/92S-9

Abstract

This research created a graphics-oriented computer program which was used as part of a Visualization, Verbalization, Algorithmization and Mathematization (VVAM) learning protocol. The curriculum for this research was the study of variance/covariance structures of bivariate and multivariate normal populations. The program displays the geometric images corresponding to the various possible covariance structures. These images facilitate and encourage experiment-based self discovery learning. The program encourages the student to take an active role in their own education. The program created is self contained, calculating all the statistical values it requires to create the various geometric images. The student has complete control in choosing what scenario they wish to study. The program was tested using four specific scenarios which represented a cross section of all possible scenarios. Within each of these scenarios were several options which were designed to encapsulate different aspects of the curriculum. The results of the research showed that the program, under the aegis of the VVAM, does facilitate the visualization and verbalization of the complex mathematical concepts associated with covariance structures. The learning environment was found to promote the creation of new knowledge on three distinct levels.

INTERACTIVE GRAPHICS SYSTEM FOR THE STUDY OF VARIANCE/COVARIANCE STRUCTURES OF BIVARIATE AND MULTIVARIATE NORMAL POPULATIONS

I. Introduction

1.1 General Issue

Is today's work force prepared to solve problems in a world characterized by increasing complexity, rapid change, unprecedented challenges and opportunities? Looking at the nation's economic status, it seems the answer is no. This lack of preparation can be linked to this country's current state of education. According to a recent *Newsweek* article, "one international study after another places U.S. school kids near the bottom of the heap in mathematical achievement" (*Newsweek*, 1990:52-54).

These poor achievement scores are just a symptom of the real problem, which is an educational system that promotes passive learning and rewards students solely on the basis of grades. This passive learning environment creates students (who later become workers) who are complacent and uncreative. What is needed is an educational process that promotes active learning. An educational process based on active learning would provide an environment conducive to students becoming creative and self-motivated.

The challenge, therefore, is to create a new pedagogy that will lead to continuous improvement in the educational system and in the work force. Revamping the entire educational process in this country is beyond the scope of this research. What can be done, however, is to change the statistical pedagogy at AFIT into a more active learning process.

1.1-1 Typical Learning Process. In the mathematical learning environment there are two contrasting approaches to teaching new concepts. Traditionally, the teacher gives a student the concepts in a rigorous mathematics language in the form of formulas. The student is then guided through an exercise of applying these formulas to standard well-defined situations or cases. The teacher then discusses the implications of each case. This process should result in the student being able to recite the formulas and concepts and being able to apply them to canned situations. It is also

supposed to lead the student toward learning the concepts in a meaningful way. This process is characterized by four steps: Mathematization, Algorithmitization, Verbalization and Visualization (MAVV) (see Figure 1.1). This approach stresses the transmission of the teacher's knowledge to

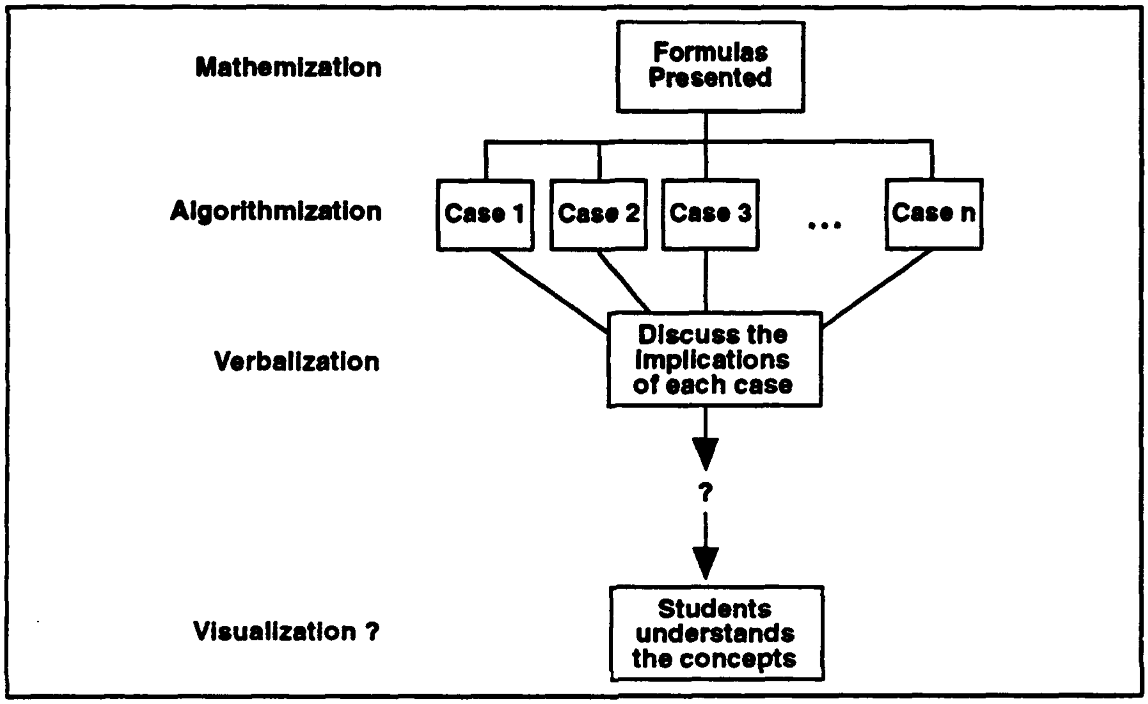


Figure 1.1. Typical Mathematics Learning Process

the student and encourages passive and uncreative behavior. Such a learning system promotes boredom and precludes stimulating the mind of the thoughtful student.

1.1-2 Contrasting Learning Process—VVAM. A contrasting process to the typical passive MAVV approach is the Visualization, Verbalization, Algorithmitization, Mathematization or VVAM Protocol (see Figure 1.2) developed by Stone W. Hansard (Hansard, 1990:24-30). In contrast to the MAVV process, the VVAM protocol stresses a student's active involvement in the learning process from the beginning. In the first step: visualization, the student must concentrate and experiment with dynamic images representing the concept the instructor is attempting to motivate. Next, the student must articulate and verbalize what he sees. Then the student, not the teacher, is asked to create an algorithm which formalizes the mathematical activity encompassed by the visualization. Finally, the student is asked to translate the algorithm into the formal language of mathematics (see Figure 1.2). The VVAM demands that a student actively

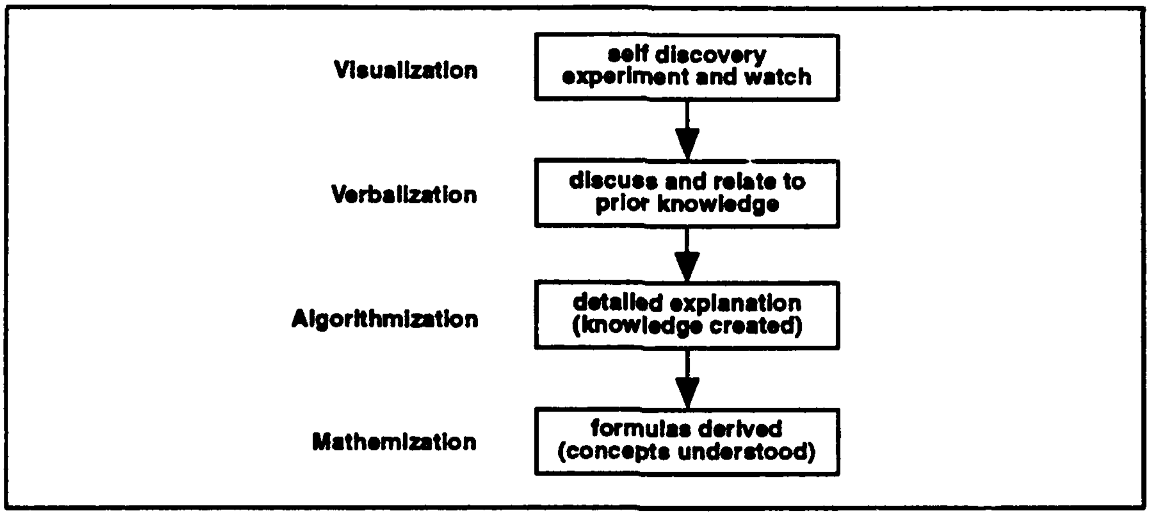


Figure 1.2. VVAM Learning Protocol

involve himself in the learning process and requires him to create knowledge he needs for himself instead of suggesting the teacher transfer such knowledge directly to the student.

The MAVV process rewards a student for producing an appropriate response to an instructor's query, while the VVAM protocol rewards a student for being an active participant in a creative and constructive learning process. The contrast between the MAVV process and the VVAM process is obvious (see Figure 1.3). The MAVV process rewards the student's ability to accept the product

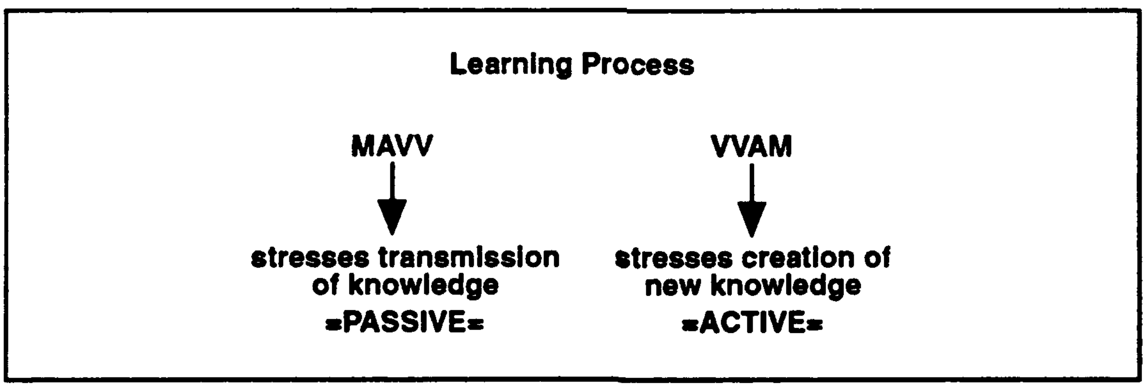


Figure 1.3. Contrasting MAVV and VVAM

(new knowledge), while the VVAM process continually rewards the student by allowing him to be creative and be involved in the learning process.

1.1-3 Schwab's Commonplaces. In order to develop and implement a new learning process, it is important to identify the key elements of any **educating event.** According to

Schwab, the learning process has four commonplaces: "the teacher, the student, the curriculum and the milieu or governance" (Schwab, 1973:502). These commonplaces can be viewed as the nodes of a multifaceted tetrahedral system comprised of the triad: student, teacher and curriculum on a single plane, and the governance, hovering above, superior and setting the standards for total system behavior. Figure 1.4 shows this interrelationship and how each element of the educating

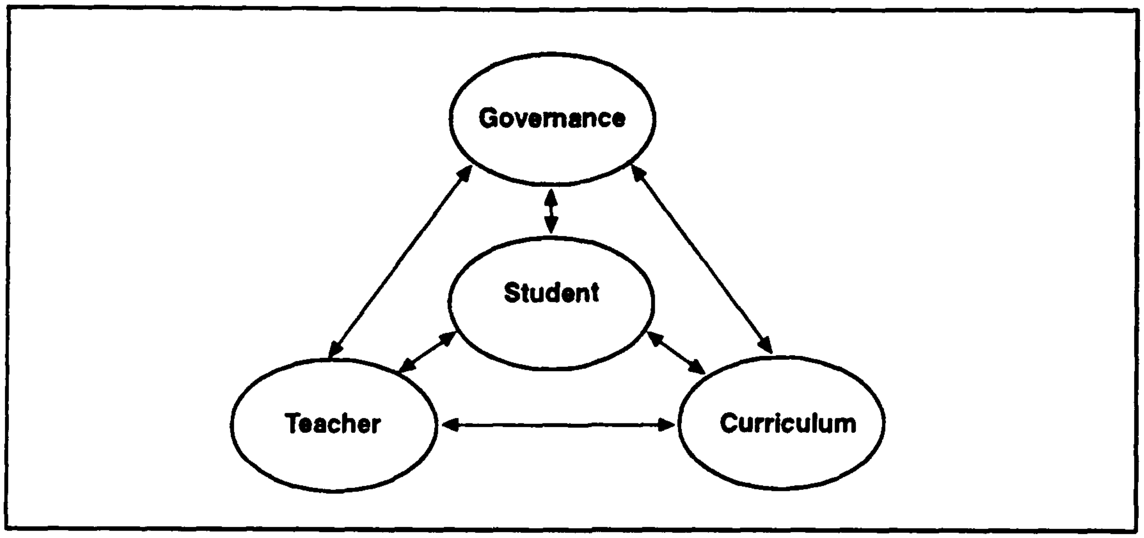


Figure 1.4. Schab's Four Commonplaces

system interacts with all other parts of the system. "It is the teacher's obligation to set the agenda, and to decide what knowledge might be considered in what sequence" (Novak, 1990:6). The student or the learner, suggests Novak,

> Must choose to learn; learning is a responsibility that cannot be shared.... The curriculum comprises the knowledge, skills, and values of the educational experience that meet criteria of excellence that make them worthy of study. (Novak, 1990:6)

Governance, or the milieu,

> is the context in which the learning experience takes place, and it influences how the teacher and student come to share the meaning of the curriculum. (Novak, 1990:6)

In most schools, teachers are competent to teach their respective subjects and students will agree that they need to take an active role in their education. And certainly, both students and teachers realize their responsibility for maintaining an interactive relationship with the curriculum. If governance is oppressive, it can hamper and even destroy the viability of such critical interde-

pendencies. The fact is, governance, whether oppressive or libertarian, is typically outside the domain of control of either student or teacher. Therefore, any pedagogy that is developed must be constructed to function within current constraints of the operative governance. The other commonplaces: the curriculum, the teacher and the student, can then be internally restructured and externally integrated to ensure that an effective learning environment manifests.

The three elements of the education event—the triad of the student, the teacher and the curriculum (see Figure 1.5) should be the focus for any pedagogical innovation.

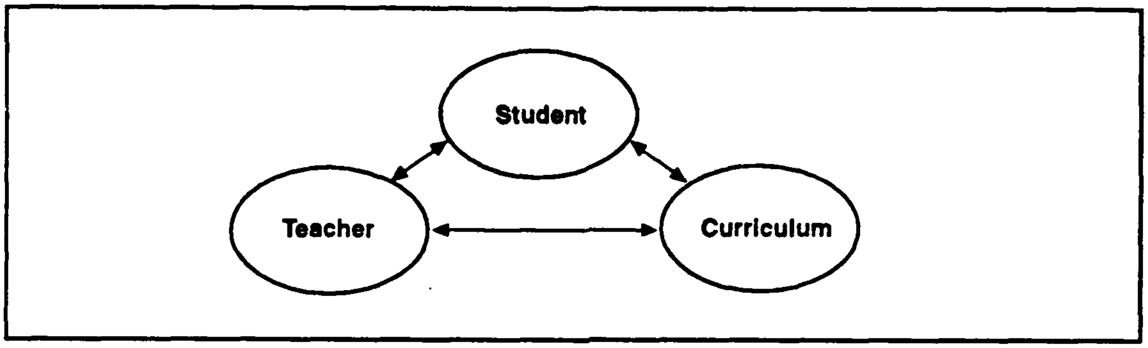


Figure 1.5. Triad of Key Elements

The teacher's role under the VVAM protocol is to create, monitor, control and enhance the learning process. The teacher should create an agenda, decide what concepts the students need to learn and, generally, suggest in what order these should be studied. Additionally, the teacher should gauge a student's ability to construct new knowledge throughout the learning process. Knowledge construction is facilitated by a teacher's constant questioning of the student. The temptation to give away answers, thus denying a student the opportunity for learning a concept on his own, should be resisted.

For the purpose of this research, the student will be working to learn the concepts of multivariate statistics and, in particular, the characteristic principles of covariance. The curriculum will be captured in an interactive graphics system in which a student can visualize and verbalize covariance structures. Once these covariance structures are comprehended, they can be used to creatively generate algorithms and formulate mathematical models of the theoretical and empirical covariance structures. This will create a rich two-way interaction between the student and the curriculum (see Figure 1.7).

The purpose of this research was to determine if a meaningful learning process could be

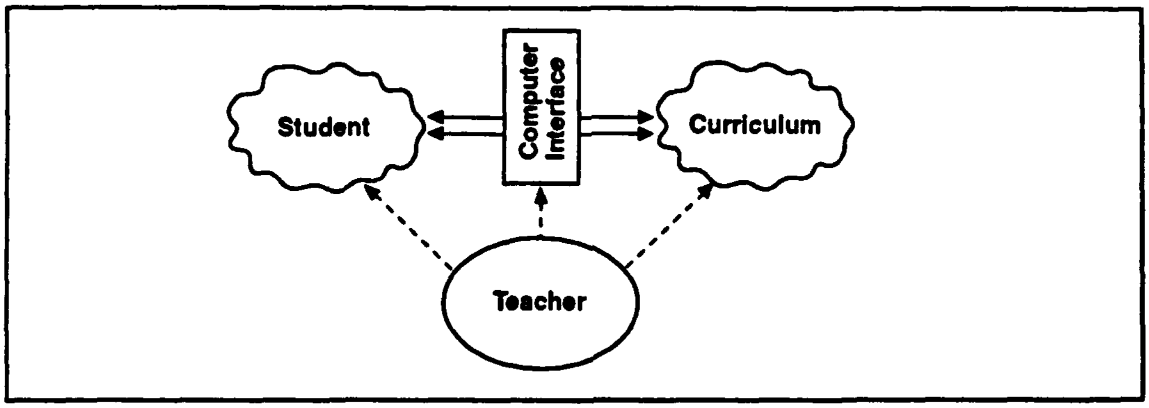


Figure 1.6. Computer Bridges Student and Curriculum

developed using a computer graphics program and the VVAM protocol created by Hansard (Hansard, 1990:24-30).

1.2 Specific Problem

The problem has two facets:

1. How can a passive book/test centered learning activity, which is primarily a rote/reception learning exercise, be transformed into an active and meaningful learning exercise where discovery learning is fostered and encouraged?
2. What is the most efficacious way to use a computer to create an interactive and graphically-based learning system?

Such questions imply the computer is the key to interaction between the student and the curriculum—guided and facilitated by the teacher. (see Figure 1.6). This was, indeed, the major stimulus for the research carried out during this thesis effort.

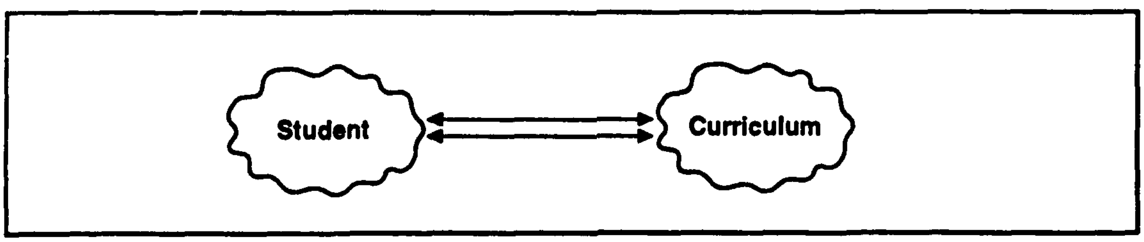


Figure 1.7. Student/Curriculum Interaction

1.3 Investigative Question

Given the problem, as stated above, it is now possible to formulate the main question posed by this thesis, mainly:

Can a personal computer-based interactive graphics system be developed and employed under the aegis of the VVAM protocol to create an imaginative, visually interactive, meaningful learning environment for the study of covariance?

1.4 Research Hypotheses

Finding an adequate answer to this investigative question involved obtaining reasonable support for the following three research hypotheses.

1. The VVAM learning protocol, developed by Hansard, can be used to orchestrate an experiment-based environment for constructively learning and actively exploring variations of the covariance structure associated with selected multivariate databases.
2. Computer-generated graphic images, embedded in an appropriately pictorialized vector space, can be employed to enhance the identification and understanding of critical statistical features traditionally used to characterize the covariance structure of bivariate and multivariate normal populations.
3. Exercising a graphics oriented program to assist the first two steps of the VVAM protocol, visualization and verbalization, and relevant heuristics for constructive learning will enable students to gain a profound and experiment-based mastery of the multivariate covariance analysis concepts.

1.5 Justification for Research

Traditionally, classroom presentations of mathematics and statistics emphasize the transmission of knowledge, formulae and techniques for conducting operations associated with descriptive and inferential procedures. Symbols are introduced for such entities as the sample mean and the sample variance as students are encouraged to rotely memorize and blindly apply formula without really understanding, or being encouraged to understand, the rationale and fundamental theory motivating the application of said formula.

Thus, learning becomes a passive, rather casual affair an exercise in arithmetic for the sole purpose of "being able to pass exams" routinely administered in the conventional math or stat course to "assess what the student has learned." After years of mastering the strategy and tactics of rote learning, some students do very well, others demonstrate average mastery of the material, while still others drop out in sheer terror or boredom sensing, somehow, such "learning exercises" ought to be offered to those who cannot, or simply do not wish to exercise their reasoning faculties.

Certainly, many teachers and many students are not content to remain under such a mindless regime. The brighter and more attentive students usually try to ask questions, seeking at least some insight into what they are being told to do to survive the course. Teachers, frustrated with such "cookbook oriented activity," make sincere efforts to explain the properties and importance of what is being covered in the course. Some even attempt a socratic dialogue, hoping that something they say will spark student enthusiasm. In such classrooms, where at least the motions of learning behavior can be observed, conversations and dialogue begin to lift the pall of ennui. Some students even begin to ask for more work than assigned or suggest they be allowed to take on "special projects." Alas, almost always, governance and the sheer magnitude of competing priorities (other classes and concurrent rote learning exercises, exams, etc) defeat the best efforts of the most aggressive and self-motivated student. After an initial sense of excitement, the learning system soon settles down into an even more befogging process as those students and teachers, who at least gave "the better way" a go, become discouraged and overwhelmed by conventional demands to achieve "high test scores" and "superior performance on timed exams."

The fact is mathematics is a language: words, terms, and symbols, and so forth that needs to be used in a creative way if the abstract representations facilitated by the syntax and structures of the language are to be meaningfully mapped to real world phenomena. Such employment of the language must be an active and creative exercise on the part of both student and teacher. New knowledge is constructed, as Ausubel, Gowin and Novak so eloquently suggest. Certainly, such knowledge construction necessarily requires rote memorization of key terms, concepts, and the ability to intelligently manipulate the entities that constitute the working elements of the language. But, such activities, as important as they are, are not, and should not be, the central focus of a course in statistics or any mathematically based curriculum. Students must be encouraged to actively use the language of mathematics to model what they observe in the real world, or their mental world, after they have a chance to see or visualize just what it is they need to model. The traditional emphasis on the transmission of knowledge and presentation of formulae with the hope that students will understand their significance at some later time should be reversed—180 degrees. First, the significance of any concept, formulation, or principle needs to be mastered. This requires an environment in which concepts and their significance can be visualized and discussed with sufficient involvement of both student and teacher to ensure the student can articulate what any concept really means: both in theory and in practice. Only then should a student be encouraged to

algorithmitize a procedure competent to implement the concept and creatively formulate a mathematical statement that symbolically captures the essence of what he now understands about the concept under study.

Students need to know they can construct whatever tools they may require—especially in the face of unprecedented problems which they will inevitably encounter during research activities and upon graduation into the greater world of military and/or industrial management. This thesis effort develops a graphically based learning environment in which students can gain such confidence and competence. It fosters an intense visual encounter with mathematics and statistics from the start of any learning exercise. Via the adoption of the VVAM heuristic it reverses the traditional practice of formalistic presentation of mathematical symbology before demands are made for meaningful comprehension of concepts. An example of such pedagogy may help the reader gain an appreciation of the emphasis and revolutionary implications of this thesis.

The concept of linear independence of vectors plays a major role in matrix algebra. Consider the task of helping students acquire the notion of what linear independence implies. One can find a multitude of math texts that give a formal and highly rigorous definition of linear independence. Indeed, that is often the way students encounter the concept in the first place. After "burning a theorem or two into their brains" (rote memorization par excellence) they are shown some mathematical formulations and manipulations that suggest how to demonstrate linear independence between two vectors. If asked at this point in time what independence means or what utility the concept may have in the practice of matrix methods most students respond with a blank bewildered look on their face. You may even hear "who cares" from some. As far as their concerned they've got the notion in memory and, if queried about it, can parrot back verbatim what they've "learned." If awarded an A for such effort—you can be assured they are already onto some new topic—confident, based on instructor feedback, they have mastered the concept.

The system designed by this thesis would suggest first a picture, an image, of two independent/dependent vectors be displayed for student observation. It would then suggest the student be asked to verbalize what characteristics he can observe about the vectors that might suggest they were independent/dependent however that might be mathematically formulated. Since geometric vectors have both magnitude and direction the student can be asked to relate the images to real world phenomena and asked to articulate what the direction and magnitude of the displayed vectors might symbolize. Once the instructor is confident the articulation of the concept is competent and

complete, then and only then should the task of constructing an algorithm to assess for independence/dependence and a formal mathematical symbolization of the concept be introduced. Why? Because once the concept is understood in a meaningful way, the symbols, the formula, and the rigor captured by succinct mathematical symbolization will mean something to the student. He will have constructed the meaning on his own, with the help of the instructor to be sure, and rather than being requested to memorize a meaningless mass of strange and sometimes intimidating set of characters (Greek or otherwise) he will value the compactness and conciseness of such linguistic tools and find it a natural act to speak about what he sees quite clearly and can articulate in his mother tongue, at will, in a more rigorous way. If not immediately, rather soon into such a learning process, he will find it more convenient to speak with his instructor in the language an instructor is rather anxious to speak—the language of his profession. And, with the recognition of his own growing competency, now in full consciousness, the student will fire up with a joy and enthusiasm for learning that simply never goes out. The flame of inspiration that drives all scholars relentlessly in their pursuit of learning can be shared with the student in a dynamic creative dialogue about the subject at hand.

1.6 Scope of Research

This thesis confines its focus to the domain of multivariate statistical pedagogy and in particular to scenarios involving the generation or collection of a single data sample. Both bivariate and multivariate covariance structures are evaluated under situations involving the estimation and/or testing of the bivariate or multivariate population mean vector.

A generic pedagogy is developed that is computer based and geometrically motivated. The thesis effort is confined to the construction of a graphically based system and a preliminary evaluation of statistical neophytes' ability to study selected concepts of covariance analysis in an interactive way under the aegis of the VVAM learning heuristic evolved by Hansard (Hansard, 1990:24-30). No formal evaluation of system performance is undertaken at this time. Therefore, any generalizations made by thesis are tentative and confined to environments similar to that created by the product of the thesis.

II. Background

2.1 Introduction

Preliminary to the development of an adequate learning system, this research effort had to establish an adequate foundation for the methodology outlined in Chapter III by completing three tasks:

1. The major elements of Ausubelian Learning Theory needed to be studied and tabulated.
2. The need for a learning system to study covariance had to be demonstrated to exist in the statistical community.

And finally,

3. The accomplishments of AFIT students Stone Hansard and Steven Pearce, both whom employed the VVAM during their AFIT thesis research, had to be reviewed.

2.2 Learning Theory

One of the keys to developing and successfully implementing the learning process developed by this thesis was obtaining a solid understanding of the basic principles of Ausubelian Learning Theory. The core elements of this theory have been documented by Joseph D. Novak and D. Bob Gowin who were students of David Ausubel. Certainly, the most relevant premise of Ausubel's theory of learning serving the purposes this thesis is the assertion that **knowledge is constructed.**

> Knowledge is constructed. That people discover knowledge is a common myth. Discovery may play a role in the production of new knowledge, but it is never more than just one of the activities involved in creating new knowledge. (Novak & Gowin, 1990:4)

In other words, "knowledge is not discovered like gold or oil, but rather constructed like cars or pyramids" (Gowin & Novak, 1990:4) Novak and Gowin suggest "The construction of new knowledge begins with our observations of events or objects through the concepts we already possess" (Novak & Gowin, 1990:4) This is exactly what occurs in the visualization and verbalization steps of the VVAM protocol.

Gowin and Novak, following Ausubel's lead, suggest any learning exercise can be represented by a coordinate pair in a 2 dimensional plot where the horizontal axis represents a continuum of learning activity from totally passive to utterly self-guided and the vertical axis represents mental activity from rote-memorization to creative and dynamic thinking. Rote and meaningful learning are contrasted in Table 2.1. Figure 2.1 suggests reception learning implies information to be learned

TABLE 2.1. Contrasting Rote and Meaningful Learning
(Novak & Gowin,1990:167)

Rote Learning	Meaningful Learning
• Arbitrary, verbatim, non-substantive incorporation of new knowledge into cognitive structure.	• Non-arbitrary, non-verbatim, substantive incorporation of new knowledge into cognitive structure.
• No effort to integrate new knowledge with existing concepts in cognitive structure.	• Deliberate effort to link new knowledge with higher order, more inclusive concepts in cognitive structure.
• Learning not related to experience with events or objects.	• Learning related to experiences with events or objects.
• No affective commitment to relate new knowledge to prior learning.	• Affective commitment to relate new knowledge to prior learning.

is selected and transferred by the instructor while discovery learning requires that the student identifies and chooses the relevant information to be learned. (Novak & Gowin, 1990:7)

Once the author of this thesis was confident that a theoretical basis to guide development of the learning environment proposed by this thesis, statistical periodicals were reviewed in search for suggestions and support for a computer-based graphical system designed to facilitate the study of covariance.

2.3 Supporting Documentation for a Geometric Presentation of Statistical Concepts

Marvin S. Margolis states in one of this papers,

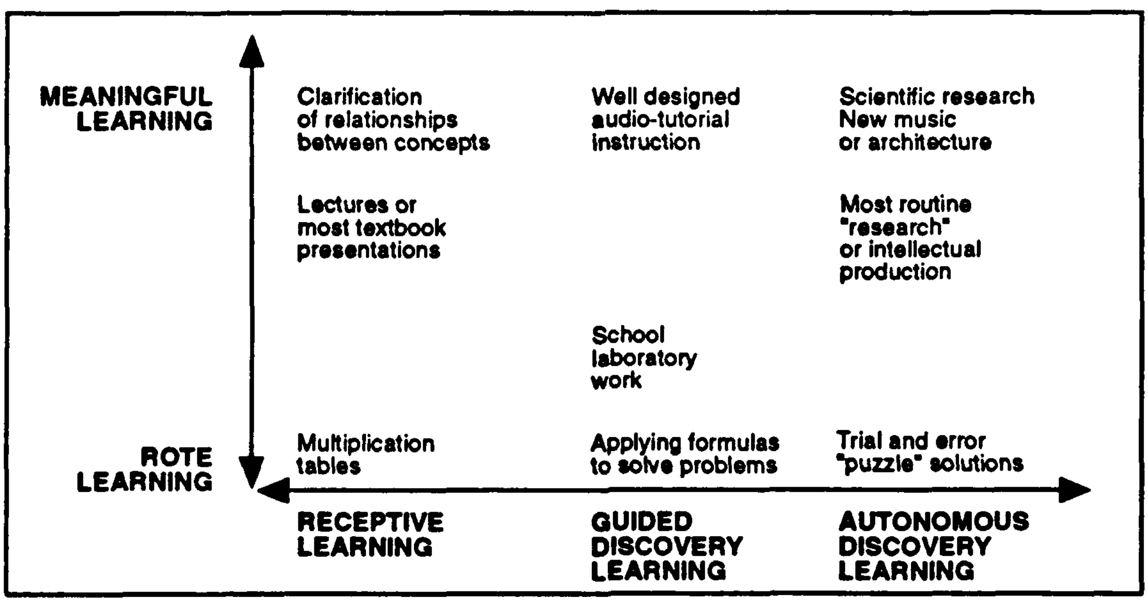


Figure 2.1. Learning Continuums (Novak & Gowin, 1990:8)

> An understanding of the geometric aspects of elementary statistics may sometimes assist a student of statistics more than elegantly derived formulas. (Margolis, 1979:131)

Margolis sees "geometric thinking...as a means of visualizing and thereby improving a student's comprehension of basic statistics" (Margolis, 1979:131). Margolis and the author of this thesis are convinced that the geometric approach to teaching statistics is a powerful tool for conveying basic statistical concepts to students (Margolis, 1979:135).

Saville and Wood point out that the main body of statistical concepts and methods are the "simple application of the mathematics of Euclidean N-demensional space" (Saville & Wood, 1986:205). They suggest that a teaching method is needed to bridge the gap between rote style methods (used in many basic statistical courses) and the extremely sophisticated derivation methods (used in higher-level mathematical based courses). They feel the method of greatest promise is the geometric approach (Saville & Wood, 1986:205). Also, Bryant, alluding to Ausubel's ideas of linking ideas within a cognitive structure, says, "Geometry seems to be the natural way to emphasize the unity of the fundamental ideas [of statistics]" (Bryant, 1984:38).

Because the author of this thesis knew a geometric approach applied to the study of covariance would require the development of a dynamic, computer-based interactive 2 and 3-dimensional display environment, a review of the work of Stone Hansard and Steve Pearce, former AFIT students who pursued such software design as part of their thesis research, was undertaken.

2.4 Previous Work with VVAM

Hansard applied the VVAM protocol to matrix algebra. His particular study did not exercise the VVAM protocol's full potential and did not facilitate a geometric approach. Steve Pearce applied the VVAM protocol to the teaching of the General Linear Model, an area of statistics that Bryant, Saville and Margolis suggested would benefit from such an approach.

Pearce demonstrated it was possible to project vectors in a 3-dimensional space that could be manipulated and viewed from various angles (Pearce, 1991:32-61) Pearce also created several software procedures that, with minor modifications, can be used to input and display various sized matrices on the computer screen. (Pearce, 1991:127-136) Unfortunately, Pearce was not able to extend his system to encompass the study of covariance and covariance structures of bivariate and multivariate normal distributions. Therefore, the author of this research decided to devote his research activity to the design and implementation of a system for the geometric study of covariance.

The following chapter documents the methodology he used to accomplish this task and provides a comprehensive review of the five major phases involved in designing, constructing and evaluating the software required to bring the system to an operational status.

III. Methodology

3.1 Introduction

Creating a meaningful learning environment for the study of variable dependencies involved the computerization of a standard MAVV driven curriculum into a VVAM governed interactive graphics system for visualizing covariance structures. The methodology used to accomplish this task is outlined by this chapter. Implementation of the research design required the completion of two research activities:

1. finding an answer to the investigative question posed by the thesis and
2. evaluating the three research hypotheses posed by the thesis.

The sample data acquired to complete these two important tasks was obtained by generating selected graphical displays during VVAM governed learning exercises in which the author of this thesis served both as student and evaluator. Formal assessment of the learning system's ability to provide a meaningful learning environment for studying covariance was accomplished by applying subjective criteria to a limited and carefully selected set of statistical scenarios. The following sections present an overview of the paradigm employed to carry out the research effort and outline the format of the graphical images produced by the graphics program. The final section documents the evaluation criteria that were used to assess how effectively the images of covariance structure produced by the program facilitated the student's ability to master the concept of covariance.

3.2 The Overall Research Design

The research design employed by this thesis was developed to ensure a meaningful learning environment could be constructed. The following process flow diagram provides a macro-level image of the overall research design. Succeeding sections of the chapter document specific steps undertaken to complete the following five phases of the research methodology (see Figure 3.1, page 3.2).

1. overall planning
2. reviewing and documenting the mathematical foundations of covariance
3. developing software to display covariance structures
4. selecting scenarios to demonstrate the ability of the software system to create a meaningful learning environment for the study of covariance
5. establishing criteria to evaluate the learning system's efficacy.

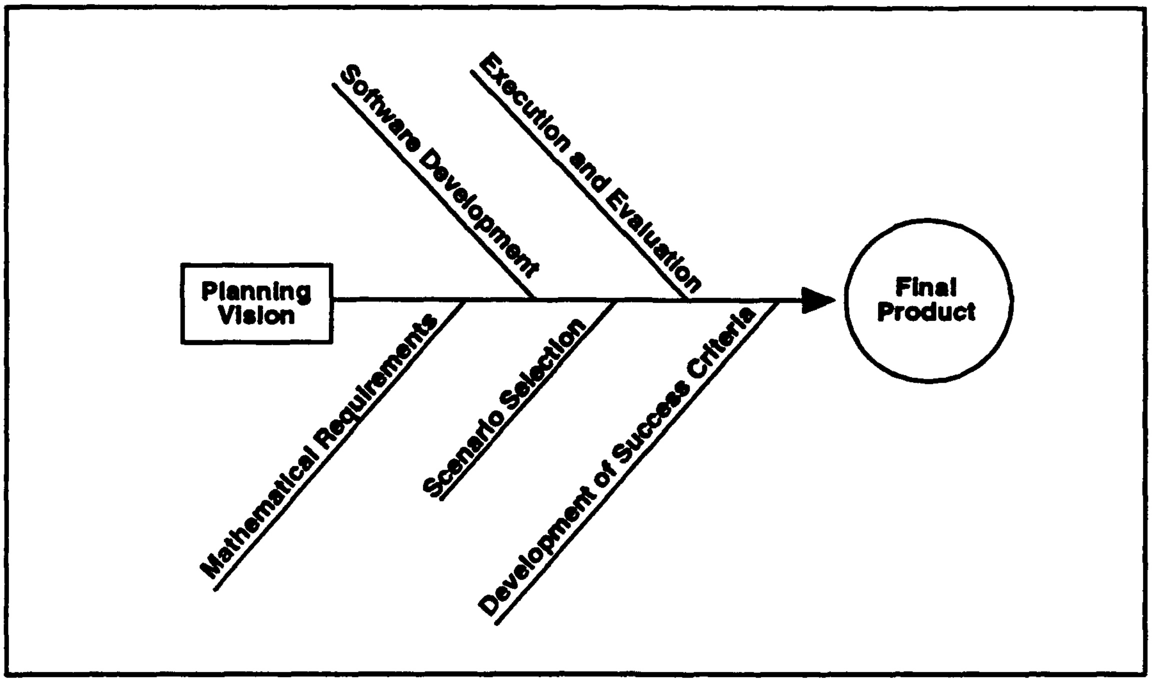


Figure 3.1. Program Development Process

3.2-1 Phase 1—Overall Planning. This turned out to be the most important phase of this thesis effort since it was at this point in the research process that the vision and goals of the thesis were established. The vision involved determining precisely what images of covariance the graphics program should produce and the steps, both pedagogic and programming, that would be required to ensure successful development of the final product: a meaningful learning system. A detailed list of program specifications was developed and ultimately consisted of the following requirements:

1. The program should display geometric images which characterized the covariance structure to the student.
2. An option to display 2-dimensional or 3-dimensional images should exist, in response to the student's choosing, to study data generated from either a bivariate or multivariate normal population.
3. Images should be capable of being rotated around all axis in order to allow the student to view any geometric image from all angles.
4. The student should be able to enlarge or shrink images, at will.
5. The screen should be organized into four subscreens. One section should be devoted to

displaying key statistical values while the other three sections should contain different graphical images.

6. The student should be able to request enlargement of any section of the three graphics screens so as to fill the entire screen with that one subscreen thus allowing a student to make a closer examination of the particular image displayed by that subscreen.
7. The program should facilitate the inputting of data from both the keyboard and from data stored on a disk.
8. The program should use different colors to highlight and distinguish critical items displayed on the larger screen or subscreens.
9. The program should display a screen of summary statistical data prior to displaying graphs.
10. The program should have the ability to be restarted quickly so that new images can be displayed before previous images are forgotten.
11. The program should calculate all required statistical values internally.

Once these requirements were delineated, a comprehensive review of the mathematical concepts involved in carrying out this covariance analysis was undertaken. The multivariate concepts required to complete this thesis effort are documented in the following section.

3.2-2 Phase 2—Documentation of Mathematical Foundations of Covariance Analysis. Real world complexity typically involves the simultaneous interaction of many factors which, when measured and evaluated as a group, requires the assimilation of a fixed number of observations on *two or more variables.*

Whenever the concern is with analyzing measurements made on several variables these measurements (or data) are usually arranged and displayed in various ways. Multivariate covariance and correlation analysis which are the main subject of this thesis require measurements to be recorded in a matrix, a rectangular array, usually symbolized by an X matrix of p rows and n columns.

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1j} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2j} & \dots & x_{2n} \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ x_{i1} & x_{i2} & \dots & x_{ij} & \dots & x_{in} \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ x_{p1} & x_{p2} & \dots & x_{pj} & \dots & x_{pn} \end{bmatrix}_{p\,x\,n} \qquad (1)$$

This array X contains the data consisting of n observations on p variables. In this thesis, analyses are constrained to the bivariate (2 variable scenario) or trivariate (3 variable scenario) since graphical displays beyond 3-dimensions are difficult, if not impossible, to display.

Descriptive statistical evaluation of such a data array attempts to summarize (graphically or numerically) outstanding tendencies of the particular sample of data: its central tendency, the individual variances of single random variables, and most importantly, the synergetic association of every pair of random variables. It is the role of the covariance matrix to succinctly represent such synergy.

This research effort formulates three categories of multivariate graphical displays. Each presumes an underlying multivariate normal population drives the data generation process. These categories of display are, respectively:

1. Theoretical/population distribution displays
2. Empirical/sample estimation displays
3. Empirical/sample testing displays.

To bring up each display one or more parametric and/or statistical entities need to be computed and analyzed. The parameters or statistics utilized by each type of display are arrayed in the following table by their domain of employment and archetypical statistical concept. Next, their formulation and significance for this study are presented. The manner in which they are employed within the computer based learning system and the more general implications of displays the software system can generate are covered in methodological and analysis discussions found in Chapters III and IV of this thesis.

3.2-2.1 Major Data Descriptors. To generate theoretical population displays or displays involving statistical inference three fundamental statistical entities must be computed via algebraic formulae or the results from the calculus of probability. The rows in Table 3.1 on the following page, list these entities by name. Column headings are employed to designate the display domain (i.e. theoretical estimation or testing) in which an entity is being used. A specific formula and a general description of each entity follow. The interested reader can review the full derivations in Johnson and Wichern's second edition of *Applied Multivariate Statistical Analysis.*

These entities serve as the critical building blocks and main input to software generating the geometrical displays of covariance structures associated with a Multivariate Normally Distributed Random Vector X: a vector whose elements are the *random variables.*

Table 3.1. Summary Table of Statistical Descriptors
Utilized During Covariance Analysis of a Single Sample of Multivariate Normal Distributed Data

Domain (Dataset)	Theoretical (Population)	Estimation (Sample)	Testing (Sample)
Entity			
Mean	$\underline{\mu}$	$\bar{\underline{x}}$	$\underline{\mu}_0$, $\bar{\underline{x}}$
Covariance Matrix	$\Sigma, \frac{\Sigma}{n}$	$S, \frac{S}{n}$	$S, \frac{S}{n}$
Correlation Matrix	ρ	R	*R*
Note: The same entities and domains are operative whether one is dealing with a bivariate or trivariate (multivariate) population and/or data sample.			

In order to document the formula for each of the entities in the three domains of interest we need to introduce an assumption that serves as the basis for all geometric displays and analyses undertaken by this thesis, mainly that $\underline{X}$ the random vector of interest is distributed multivariate normal with p random variables and covariance matrix Σ.

that is, $\underline{X} \sim N_p(\underline{\mu}, \Sigma)$

$$\text{where } f(\underline{X}) = \frac{1}{(2\Pi)^{p/2}|\Sigma|^{1/2}} e^{-(1/2)(\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})} \tag{2}$$

It then follows under the theoretical domain that

$$\underline{\mu} = E(\underline{X}) \tag{3}$$

and

$$\Sigma = E[(\underline{X} - \underline{\mu})(\underline{X} - \underline{\mu})'] \tag{4}$$

Methods of calculus can be used to obtained actual values for $\underline{\mu}$ and Σ when a specific $N_p(\underline{\mu}, \Sigma)$ is presumed. Whenever the theoretical covariance matrix of the sample mean vector $\bar{\underline{X}}$ is required, one merely divides each element of the covariance matrix of X by n (the sample size).

The population correlation matrix ρ is computed by setting up what is known as the Standard Deviation Matrix

$$D^{1/2} = \begin{bmatrix} \sqrt{s_{11}} & 0 & \dots & 0 \\ 0 & \sqrt{s_{22}} & \dots & 0 \\ . & . & \dots & . \\ . & . & \dots & . \\ . & . & \dots & . \\ 0 & 0 & \dots & \sqrt{s_{pp}} \end{bmatrix} \tag{5}$$

and executing the following matrix computation

$$\rho = (D^{1/2})^{-1}\,\Sigma(D^{1/2})^{-1} \tag{6}$$

As will be shown later in Chapter III, these theoretical entities can be used to generate a multivariate normal deviate vector of values as many times as desired to simulate a random sample of n multivariate normal deviates. Via eignvalues analysis (see below) the theoretical covariance structure represented by the Σ matrix can be obtained and graphed. Chapter IV reviews exactly how the software generated by this thesis effort accomplishes this.

It is important to note at this point that the covariance structure which is captured by the covariance matrix and which is associated with an elliptical display to be discussed directly, is the key to understanding the type, depth, and strength of synergy fostered by a dependent set of random variables.

During estimation exercises statistical estimates of these theoretical entities must be obtained. The sample mean vector is easily computed as follows:

$$\underline{\bar{x}} = \frac{1}{n}\underline{1}\;\; X \text{ where } X \text{ is a } p \times n \text{ data matrix} \tag{7}$$

The sample covariance of X presumes a data matrix of dimension $p \times n$ has been obtained and is computed as follows:

$$S = \frac{1}{n-1} X\left[I - \frac{1}{n}(11^T)\right]X^T \text{ where } x \text{ is a } p \times n \text{ matrix} \tag{8}$$

The sample covariance matrix of $\underline{X}$ is computed by dividing each element of the sample covariance matrix of X by n.

Obtaining the sample correlation matrix requires computation of the $p \times p$ inverse of the sample standard deviation matrix

$$D^{-1/2} = \begin{bmatrix} \frac{1}{\sqrt{s_{11}}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{s_{22}}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{s_{33}}} \end{bmatrix} \tag{9}$$

followed by the matrix computation of R as follows:

$$R = D^{-1/2} S D^{-1/2} \tag{10}$$

A quick review of Table 3.1 will indicate that the only remaining entity to be defined is $\underline{\mu}_0$, which is simply the plausible value for the mean vector of the theoretical multivariate population assumed to be operative under the null hypothesis when a test is conducted about a population mean vector.

3.3 Geometric Evaluation of Multivariate Normal Populations

Geometric structures displayed by the software developed by this thesis are plotted in 2- and 3-dimensional vector spaces because geometric images visually and rigorously characterize the extent and strength of relationships between a set of multivariate normally distributed random variables. Since the rest of Chapter III will outline the use and value of such images this section offers abbreviated mathematical formulations for any ellipsoid of p dimensions. Such ellipsoids are employed to create

1. An image of the theoretical covariance structures associated with a given $N_p(\underline{\mu}, \Sigma)$ population (ref: Theoretical/Population Distribution Displays)
2. An image of the empirically derived covariance structures associated with a sample from an unknown $N_p(\underline{\mu}, \Sigma)$ population and associated confidence region for estimating the unknown mean vector $[\underline{\mu}]$ or testing the plausibility of a given mean vector $[\underline{\mu_0}]$ (ref: empirical/sample estimation displays and empirical/sample testing displays.)

Contours of constant density of the p-dimensional normal distribution are ellipsoids defined by the equation:

$$(\underline{x} - \underline{\mu})'\Sigma^{-1}(\underline{x} - \underline{\mu}) = c^2 \tag{11}$$

centered at $\underline{\mu}$ with axes $\pm c\sqrt{\lambda_i}\, e_i$, where λ_i is the ith eignvalue and e_i is the ith ortho-normal eignvector (one of p orthogonal vectors, each of unit length). In this thesis the covariance matrix is used to obtain eigenvalue/eigenvector pairs.

The software requests the user enter a value of c. To obtain the desired value the user must select an α value to obtain a χ^2 percentile that can then be used in the following equation:

$$c = \sqrt{\chi^2_p(\alpha)} \quad p = \text{degrees of freedom} \tag{12}$$

If this c value is entered a contour is plotted that contains $(1-\alpha) \times 100\%$ of the probability under the relevant multivariate normal density.

When inferences about a mean vector are in order, whether for purposes of estimation or testing, an ellipsoid representing a $100(1-\alpha)\%$ confidence region can be generated by noting that

$$P\left[n(\underline{\bar{x}} - \underline{\mu})'S^{-1}(\underline{\bar{x}} - \underline{\mu}) \le \frac{(n-1)p}{(n-p)} F_{p,n-p}\right] = 1 - \alpha \tag{13}$$

where the values of $\underline{\mu}$ and Σ are unknown. Clearly,

$$n(\underline{\bar{x}} - \underline{\mu})'S^{-1}(\underline{\bar{x}} - \underline{\mu}) \le c^2 \tag{14}$$

where

$$c = \sqrt{\frac{p\,(n-1)}{(n-p)} \cdot F_{p,n-p}(\alpha)} \tag{15}$$

The software asks for a c value and will generate a confidence region that can be used to suggest an upper bound on the range of the true population vector or to determine whether or not the presumed mean vector [μ_0], employed in the testing scenario, is plausible or not.

For a fuller presentation of the mathematical rationale and inferential use of ellipsoids in various types of multivariate data analyses the reader is referred to the Johnson and Wichern text. The remainder of Chapter III and IV demonstrate how such geometric structures can be employed to create a meaningful learning environment for studying covariance. They provide examples of how visualizations of covariance structure can stimulate questions about the synergy produced by a set of dependent normal random variables.

3.3-1 Phase 3—Software Development. Software development of the covariance structure display system involved completing two distinct activities. First, an organized and logically nested set of computer algorithms had to be designed, logically evaluated for consistency and completeness and nested together to form a complete programming system for generating graphical images of covariance structures. The second step involved coding, debugging, and testing the operational competency of the programming system to facilitate a meaningful learning system under the aegis of the VVAM protocol.

3.3-1.1 Software Development: Activity 1—Algorithm Development. In order to complete design specifications for the basic algorithms and supporting routines of the programming system, a complete set of mathematical formulae had to be rigorously defined. A MathCAD 3.1 template was used to design and debug algebraic representations for all formulas that would be needed to compute specific covariance statistics and graphs of related covariance structures in the final coded program. (© 1991 Math Soft Inc.). MathCAD is a real-time personal computer-based program for creating formulas, numbers and graphics as well as text for documentation and explanations. MathCAD's unique abilities are perfect for use as an algorithm development tool. It has the ability to display formulas in the same exact mathematical representations found in mathematical text books (see Figure 3.2 on the following page). MathCAD has a built-in symbolic processor which can be used to solve equations and calculate derivatives and integrals. It also can calculate eigenvalues and eigenvectors, which are two of the most vital statistical entities used by the programming system produced by this research effort. MathCAD, itself, possesses graphics

Math Book:

$$B = \frac{3(\lambda^{1/2})}{y}$$

In MathCAD:

$$B = \frac{3(\lambda^{1/2})}{y}$$

Figure 3.2 MathCAD Formulas

capabilities that facilitated graphical algorithm testing by producing rough sketches of graphs that were ultimately generated by the final graphics package. MathCAD proved to be a very valuable debugging tool. A copy of the MathCAD template created during the design and debugging of systems software is included as Appendix H.

3.3-1.2 Software Development: Activity 2—Coding process. The coding process required three sub-activities: 1) selection of a programming language; 2) defining the hardware and operating system requirements for the host computer; and 3) outlining the system flow and the actual coding of the program.

The programming language needed to generate the graphical images required by program specifications had to possess extensive built-in graphics capabilities as well as the ability to expedite mathematically intense programming activities. The language chosen was Turbo PASCAL 6.0 (©1987, 1990 Borland International Inc.).

Turbo PASCAL is a highly structured language that uses and links units(sub-programs), functions, and procedures. Turbo PASCAL contains a built-in graphics unit with over 50 graphics routines and supports the following graphics drivers: CGA, MCGA, EGA, VGA, Hercules, AT&T 400 line, 3270 PC and IBM8514. It is a PC based language with an excellent editor and integrated development environment (IDE).

To enhance PASCAL's already extensive graphics routines, the 3-D graphics library Acromolè was used. Acromolè (version 1.0 © 1991 David B Parker) contains 38 subroutines that support high speed, real-time 3-D graphics. This package facilitates the rotation and movement of graphics images.

The hardware system targeted for this software development was an IBM or compatible

personnel computer with a hard drive, at least an EGA graphics driver and running DOS 2.0 or higher operating system.

The complete listings of program units developed for this effort are included as Appendices B-F. The overall organization of the program units is shown in Figure 3.3.

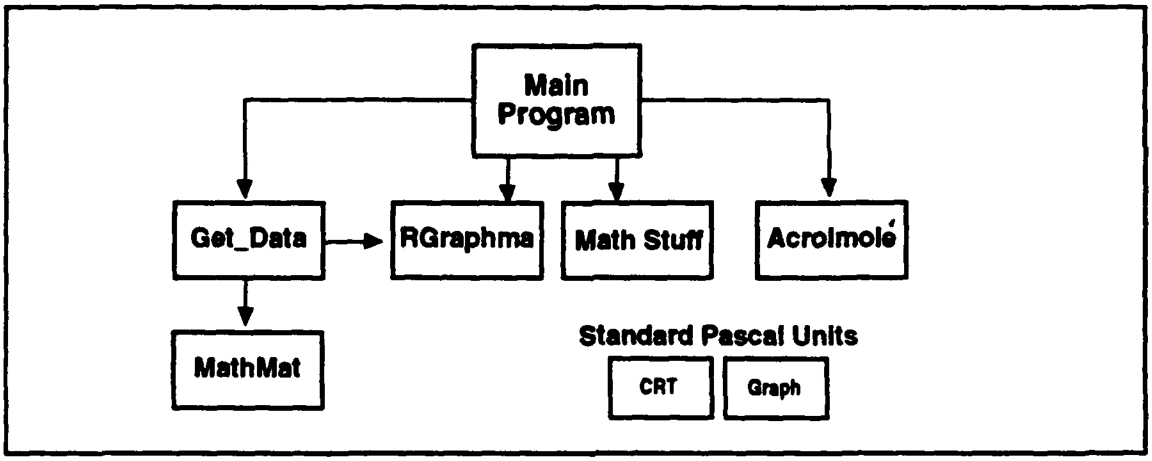


Figure 3.3. Program Organization

3.3-1.3 System Flow. The program flow can be divided into four subsystems (see Figure 3.4). Subsystem 1: Welcome Display and Request for Inputs. Subsystem 2: Statistical Entities Display Subsystem 3: Display of Images of Covariance Structure Subsystem 4: Monitoring and Responding to Inputs.

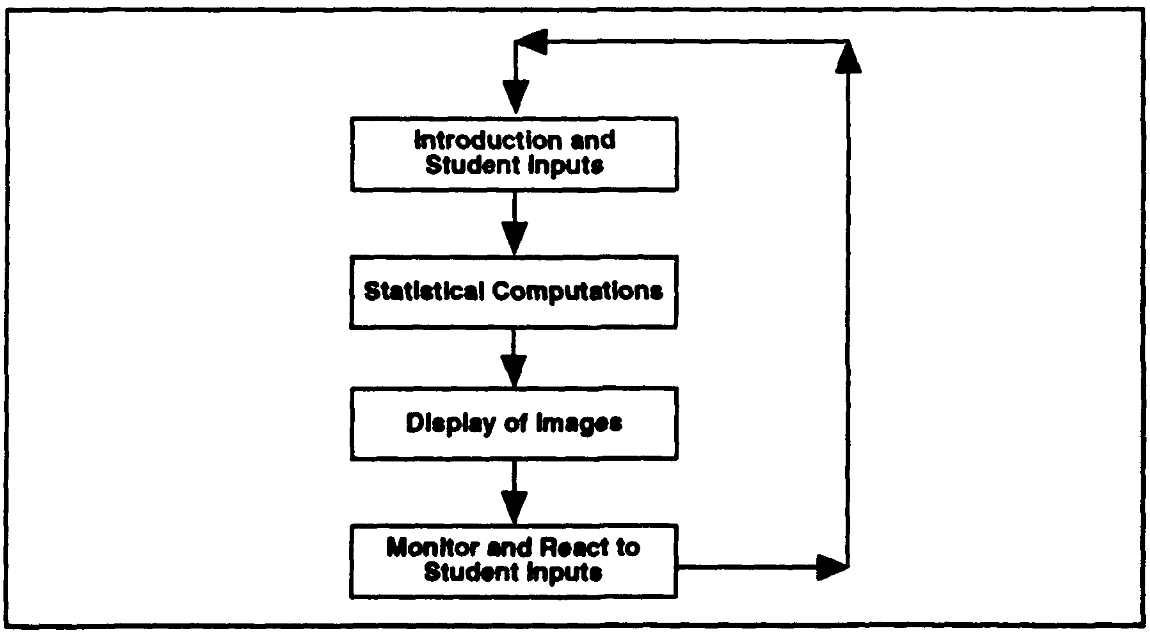


Figure 3.4. Overview of Program Flow

3.3-1.3.1 Subsystem 1—Request for Inputs. The introduction and student input section can be separated into three areas (see Figure 3.5). They are:

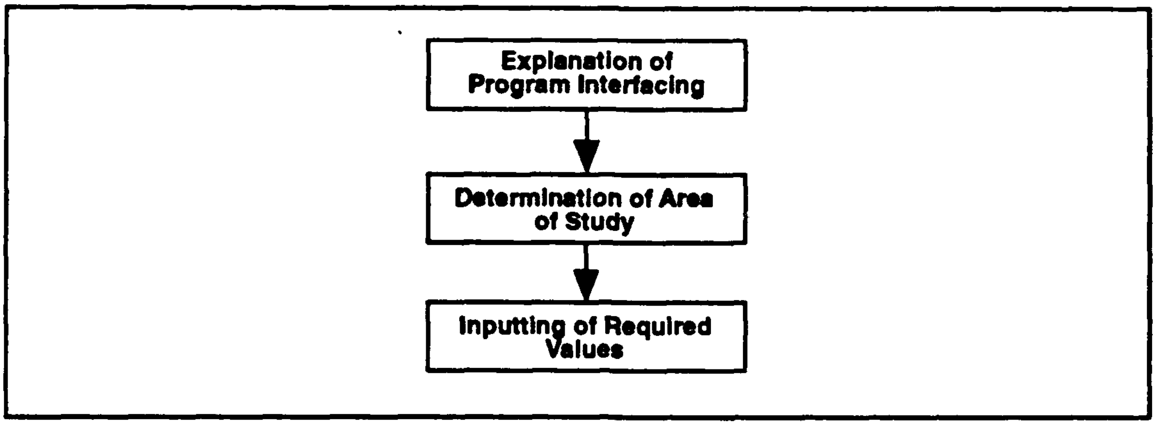


Figure 3.5. Introduction and Student Inputs

1. The explanation of program interfacing. This consists of the initial screen which explains to the student how to interface with the program and how to activate some of the special functions (see Figure 3.6 on the following page). This is accomplished by the main program unit.

2. The determination of the area of study. This is where the student answers two separate questions which will determine the type of problem they want to view. The first (Figure 3.7, page 12) allows the student to choose between the bivariate or the multivariate normal distribution. This is basically a choice between viewing a 2-D or 3-D vector space. The second question (Figure 3.8 page 13) gives the student five options which are combinations of dealing with theoretical or empirical data and data being entered via keyboard or from a file on disk. These two questions give the student 20 separate options to chose from. This is accomplished by the main program unit.

3. Inputting of required data. Depending on which of the five options the student selects they will be required to input certain values.

   a. For option 1, the strictly theoretical case they will be required to enter: $\underline{\mu}$, Σ, the *c* value, the sample size and the correlation matrix (see Figure 3.9, page 13).

   b. For option 2, the empirical case where data comes from a file, the student will have to enter; the *c* values and the sample size (see Figure 3.10, page 14).

   c. For option 3, the empirical case where data is inputted via the keyboard the student enters: a *c* value, the sample size and the sample *X* matrix (see Figure 3.11, page 14).

   d. For option 4, the empirical case that also performs a test of the mean vector, the student inputs

Interactive Graphics System for the
Study of Variance/Covariance Structures
of Bivariate and Multivariate Normal Populations

Written by Ronald G. Garlicki

Some Helpful Information

F10 *Selects current matrix or value*

While Viewing Graphics Page

F1 *Enlarges Screen #1*

F2 *Enlarges Screen #2*

F3 *Enlarges Screen #3*

F9 *Restarts Program*

ESC *Restarts Multiple Screen Graphics Page*

CONTROL
BREAK *Ends Program*

Press ENTER *to Continue*

Figure 3.6. Opening Screen

the same values as for option 2, but also the null hypothesis $\underline{\mu}_0$ they want to test (see Figure 3.12, page 15).

e. The final option, the case that compares theoretical and sample data and performs a test of the mean vector, requires input concerning: $\underline{\mu}$, Σ, ρ, $\underline{\mu}_0$ (the null), the c value and sample size. The sample data itself is read from a file on disk (see Figure 3.13, page 15).

Do you wish to deal with the Bivariate of Multivariate Normal Distribution?
Input 2 for BVN (2-D or 3 for MVN (3-D)

Figure 3.7. 2-D or 3-D

This Is the Case of Bivariate Normal Distribution:

(1) *Theoretical*

(2) *Empirical (data from disk)*

(3) *Empirical (data from keyboard)*

(4) *Test of Mean (data from disk)*

(5) *Comparision of Theoretical and Empirical (data from disk)*

Figure 3.8. Theoretical Empirical Choices

All data are inputted and displayed in matrix form as it would appear in a mathematics text book. The inputting of data is conducted by the program unit Get_Data. Once the student has inputted all needed values the next phase, statistical computations section, of the program begins.

3.3-1.3.2 Statistical Entity Computation and Display. The statistical computations section is divided into three parts (see Figure 3.14, page 16.).

1. calculation of summary and statistical values; 2. calculation of eigenvalues and eigenvectors; and 3. calculation of values needed to generate the graphical images.

Calculation of summary statistics is performed by the units Get_Data and MathMat (matrix

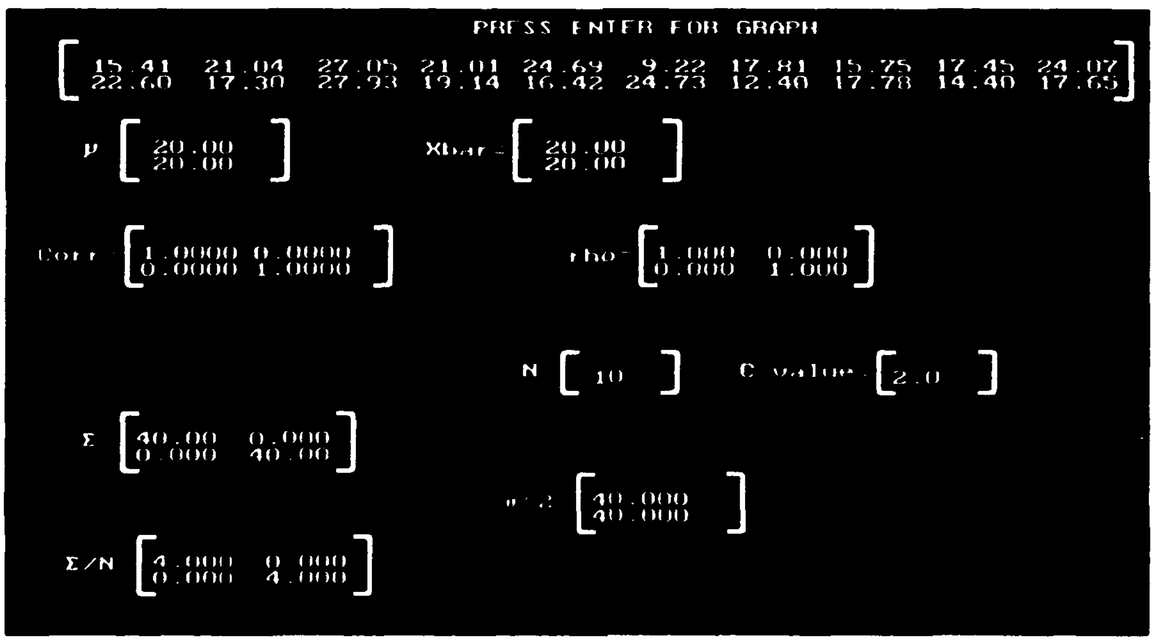


Figure 3.9. Input Screen for Case 1 Option 1: Independence

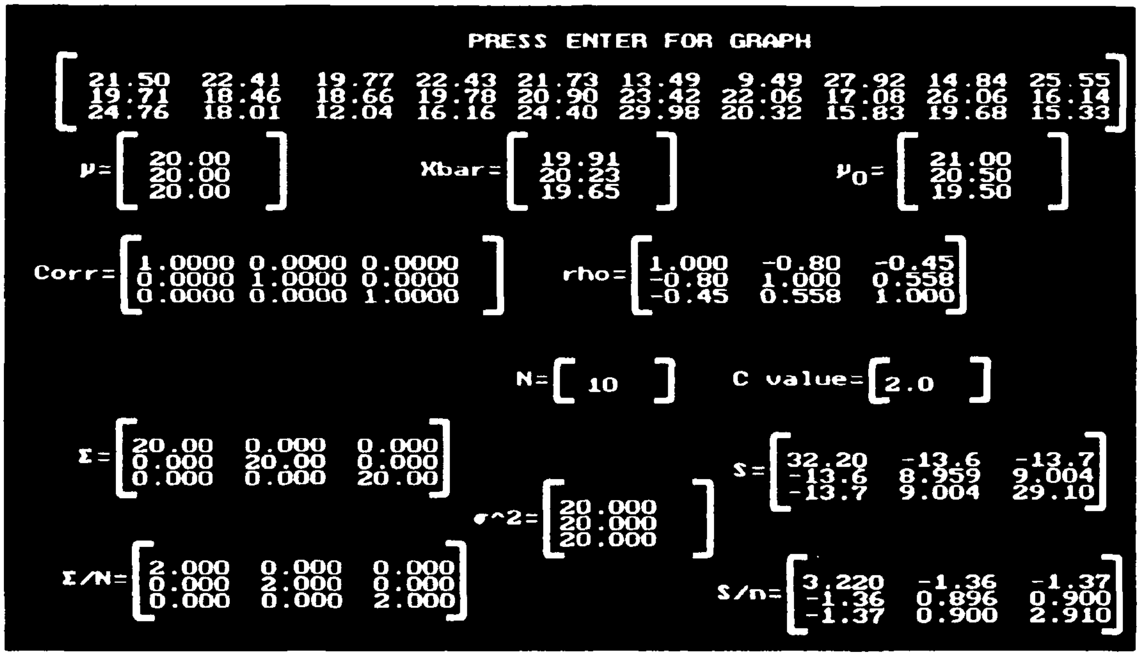


Figure 3.10. Input Screen for Case 2 Option 1

algebra functions). Depending on what option is selected by the student, different values are calculated. These values include calculations of $\bar{x}$, S, $\frac{S}{n}$, R and if theoretical information is given then a sample of random deviates from the defined population is created. These summary statistics are displayed on the input screen for the student to study prior to seeing the geometric images. As

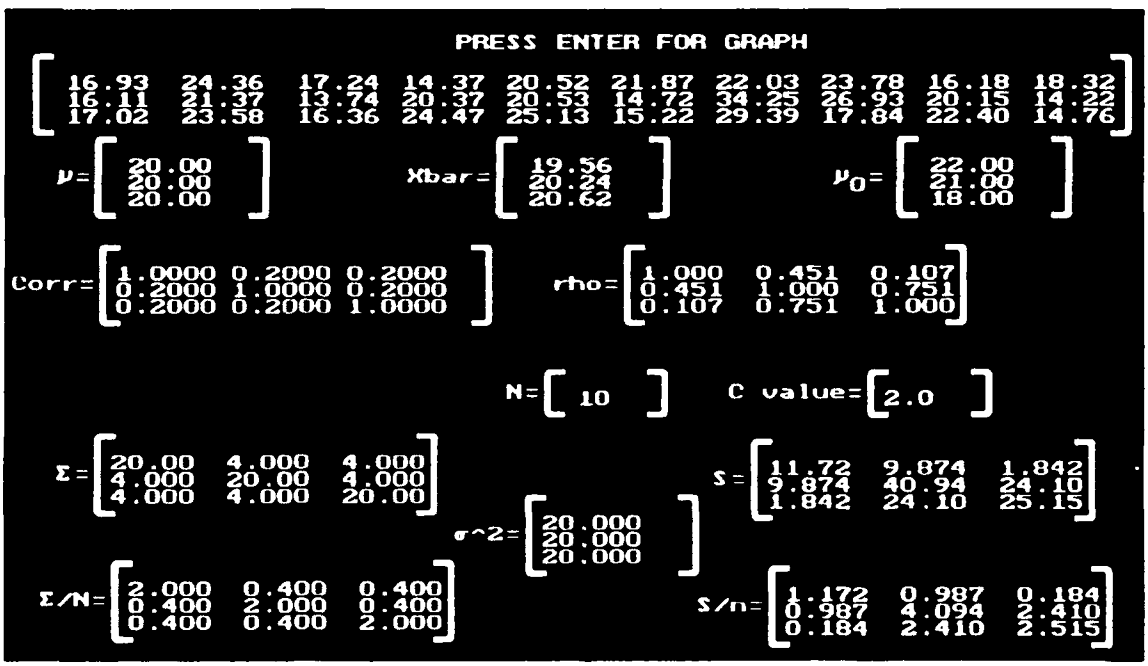


Figure 3.11. Output for Case 2 Option 2A

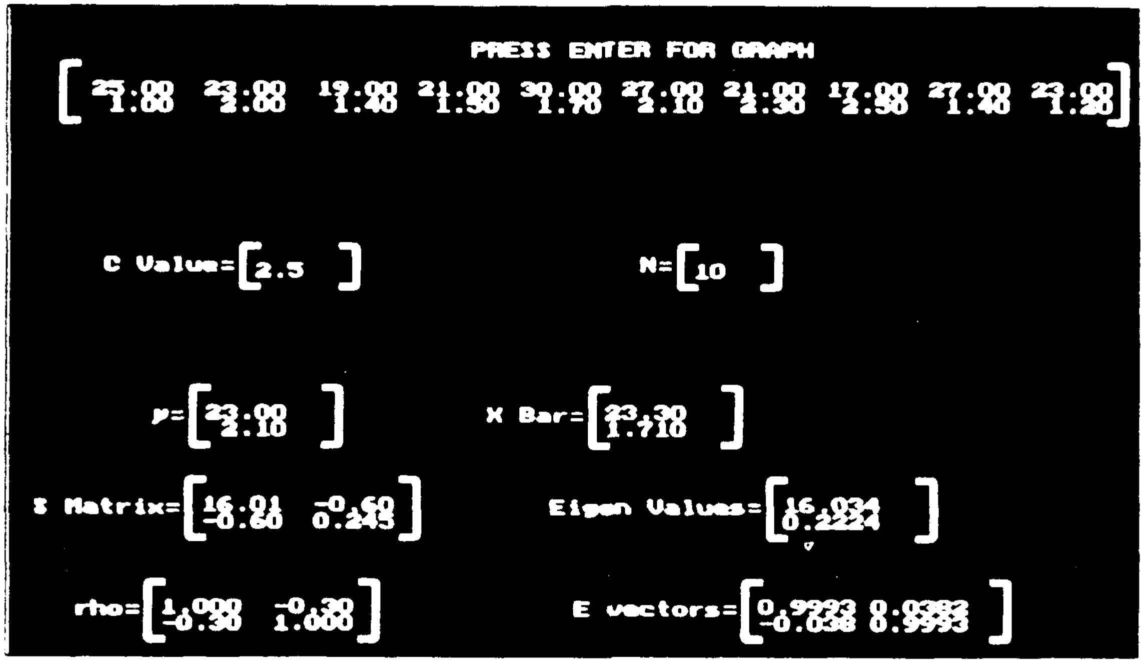


Figure 3.12 Sample Input Screen for Option [illegible]

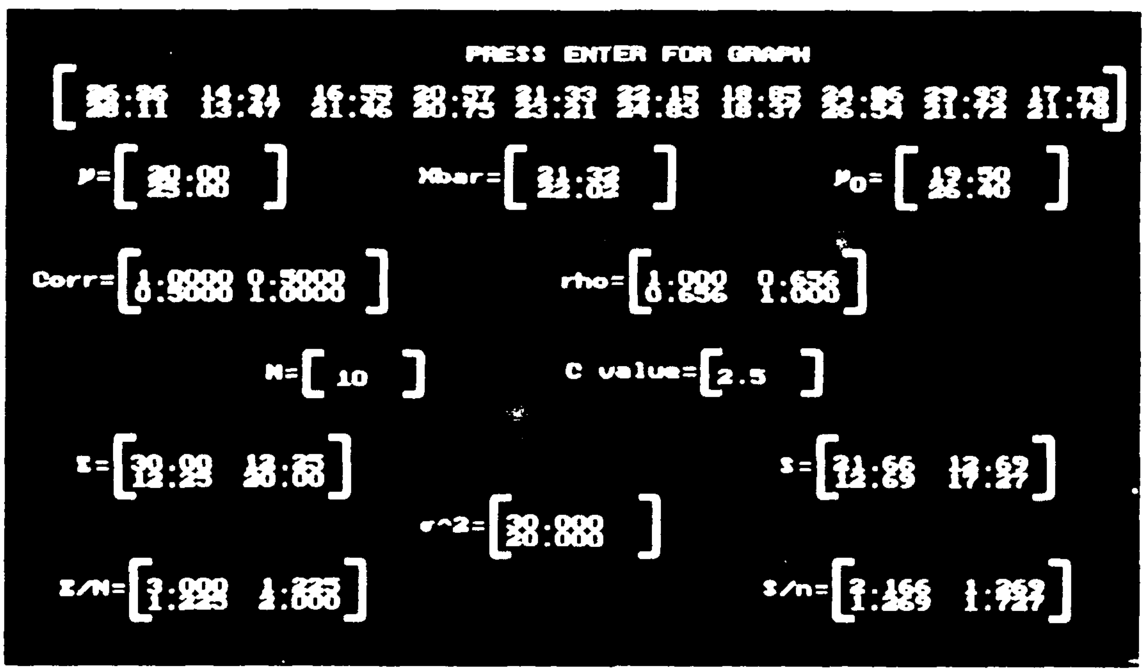


Figure 3.13 Sample [illegible]

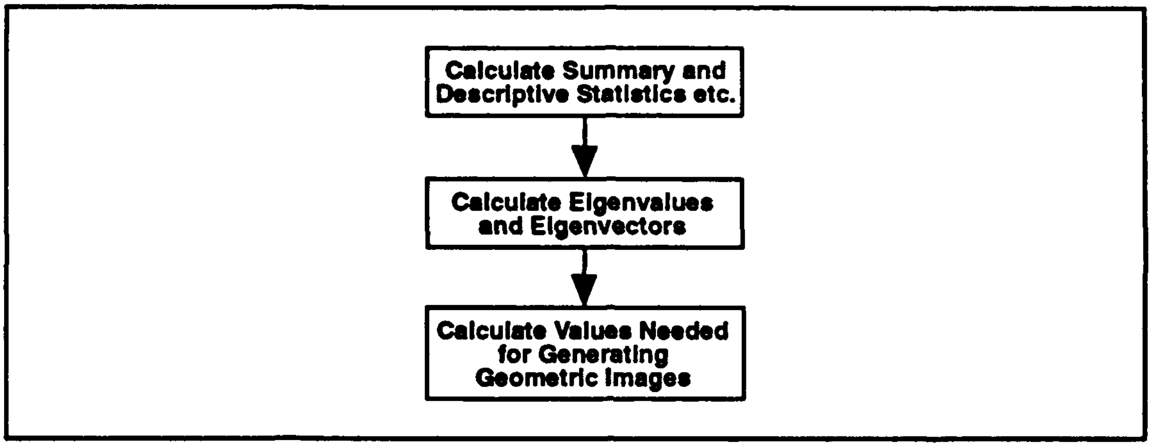


Figure 3.14. Statistical Computations

Get_Data. The Jac procedure is based on the Jacobi algorithm explained in the book *Numerical Recipes in Pascal, the Art of Scientific Computing* by Flannery Press, et al. The book contains a version of the algorithm written in PASCAL which this writer was able to use with only minor modifications (Press, 1989:387-388). It was critical to be able to calculate the eigenvalues and eigenvectors internally since they encapsulate the needed information concerning statistical distances, which govern the appearance of the geometric images. See Appendix G for further information in this area. The eigenvalues and eigenvectors are used to determine the length and endpoints of the axis of the geometric images (ellipses if 2-D and Ellipsoids if 3-D).

Calculation of the values which are needed in order to produce the geometric images is conducted in the Unit Get_Data. These values, as stated above, include determining the lengths of the axis and their endpoints. Other values include determining the domains of the relationships which define the ellipses displayed. At this point the geometric images are ready to be created and displayed.

3.3-1.3.3 Display of Covariance Structures. Once statistical computations are completed, the program begins the display phase. The display phase of the program is subdivided into four sections (see Figure 3.15 on the following page).

1. setup of the major and minor axis of the ellipse or ellipsoid;
2. calculation of the individual points of the images;
3. setup of the screen layout and;
4. the actual display.

Setup of axis and coordinate system is performed by the procedure GenerateAxis in the main

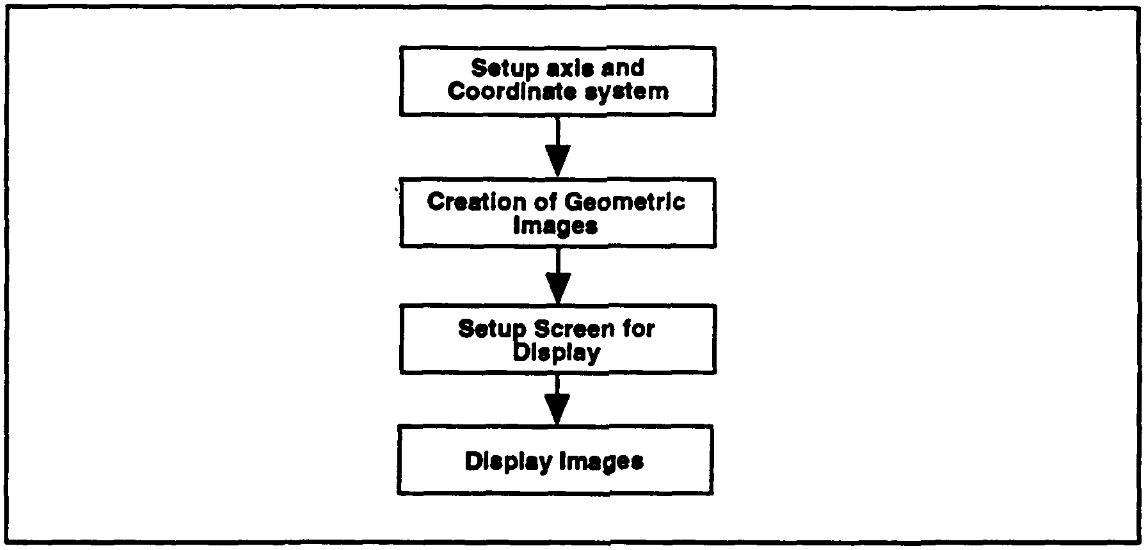


Figure 3.15. Display of Images

program unit. This procedure creates the major and minor axis of the ellipses and the x_1, x_2, and, if 3-D, the x_3 coordinate axis. This requires transforming, in some cases, 3-D information into a format which can be displayed on a 2-D computer screen without the loss of information. The transformations from 3-D images to the 2-D computer screen are completed with the aide of the Acromolè routines.

Creation of the geometric images consists of creating the points which will be displayed. The 2-D and 3-D images are based on ellipses whose relationships are captured in the procedures contained in the program unit MathStuf. There is no ellipse function in PASCAL which can be used in conjunction with Acromole. Therefore, the ellipses are created by plotting a large number of points based on the domain and characteristics of the ellipse as determined by the summary statistics and the eigenvalues and eigenvectors. The ellipsoids are characterized by three ellipses. There is one ellipse in each plane of the vector space. These planes are the x_1, vs. x_2, the x_3 and the x_2 vs. x_3. If the case being studied contains a test of the mean vector then a point is plotted at the location of the null hypotheses.

Setup of the screen is conducted in the main program. This consists of dividing the screen into four sections. One section contains the summary statistics required by the student to construct new knowledge. The other three section are reserved for different images. Each of the display screens are labeled as to what they contain.

Display of the images is performed by the main unit and a modified Acromolè unit called Utility.

The images are displayed so that the centroid of the ellipse in each screen is the center of rotation. When two or more ellipses are put in the same screen they are displayed in different colors for clarity.

3.3-1.3.4 Monitoring of Inputs. The final phase of the program is the monitoring and reacting phase. This is were further inputs from the student are read and the appropriate action is taken.

This phase of the program is a continual monitoring function with two major paths of action (see Figure 3.16).

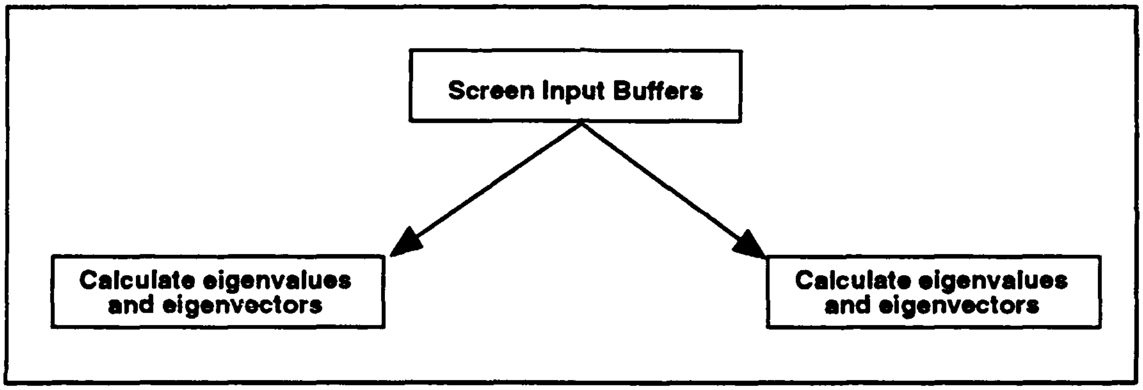


Figure 3.16. Monitoring and Updating

1. manipulation of current images and screens and
2. resetting and restarting of the program.

The monitoring consists of periodically checking the input buffer for student inputs. This is accomplished by the unit Utility. The student has several options available in regards to the images. All images have 6 degrees freedom of rotation. Rotation of the images is controlled by the arrow keys and the page-up page-down keys. The images can also be move toward and away from the student by use the minus or plus keys respectively. If the student wants to focus their attention on one of the three image screens they can press the function key corresponding to that screen (e.g., if the student wants to focus on screen 2 they press <F2>). That screen will then be enlarged to fill the entire computer screen (see Figure 3.17 on the following page). Pressing the <ESC> key will return the student to the multiscreen display.

The other possibilities for the student are to restart or end the program. Pressing the <CNTRL> and <BREAK> keys simultaneously will end the program. Pressing the <F9> key will signal the

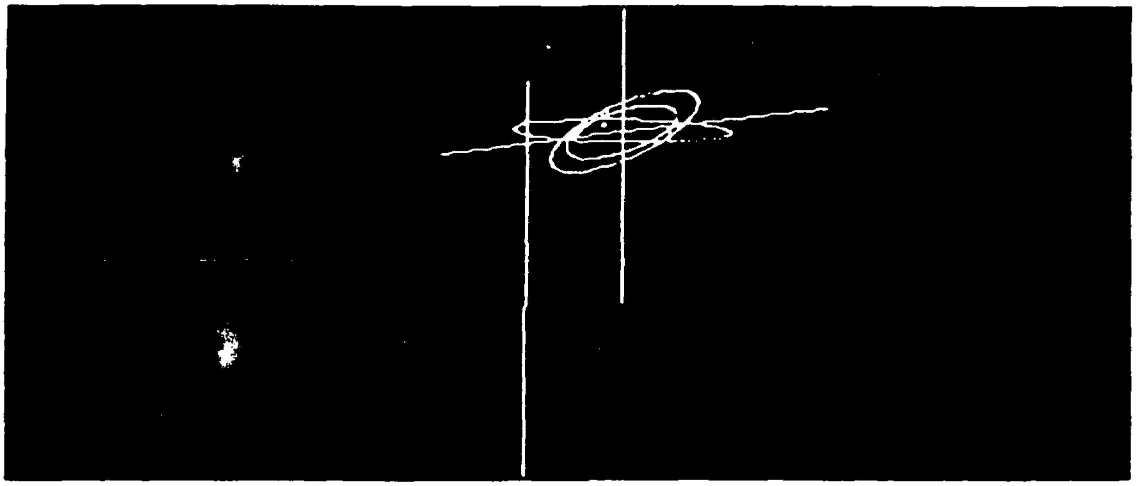


Figure 3.17. Sample Screen Focus on Screen 3

computer that the student wants to view another case. In this case the computer resets all pointers and clears the screen and restarts the program. The introductory screen is not shown when a restart occurs.

3.3-2 Scenario Selection. The scenarios selected to test the programs ability to facilitate a meaningful learning experience represent only a few of the total possible scenarios which can be examined with this program. The cases which were selected were then divided into 4 or 5 options. These options demonstrate a cross-section of the possible statistical variations which can occur within the case. Table 3.2 on the following page shows all the possible scenarios, the scenarios selected for this research and the options examined within those scenarios. What should be kept in mind when reviewing this table is that the study of covariance involves the segregation of the total or generalized variance of the distribution into its parts. The general variance can be segregated into each variables covariance with itself, also know as its variance, and into each of the variables pairwise covariances. The options within the cases were chosen to demonstrate the effect that each of these covariances has on the total variance and on the geometric images (ellipses and ellipsoids) displayed.

When studying the tables describing the scenarios and options, keep in mind the following conventions which were used as to determine what constitutes "low," "medium" and "high" correlation. When a correlation is set at a "low" value, this is a value between 0.0 and 0.4. A "medium" value is one between 0.4 and 0.6. A "high" value is one between 0.6 and 0.9. Values greater than 0.9 but less then 1.0 are considered "very high."

TABLE 3.2. Possible Scenarios

		Theoretical		Empirical		Theoretical and Empirical	
		BVN	MVN	BVN	MVN	BVN	MVN
Generate	I						
	II						
	III						
	IV						
	V						
Input	Manual						
	File						
Generate and Input	File						

		σ^2	ρ
I		$\sigma_1^2 = \sigma_2^2$	$\rho = 0$
II	A	$\sigma_1^2 < \sigma_2^2$	ρ = Low
	B	$\sigma_1^2 = \sigma_2^2$	ρ = High
III	A	$\sigma_1^2 > \sigma_2^2$	ρ = Constant
	B	$\sigma_1^2 < \sigma_2^2$	ρ = Constant
IV	A	$\sigma_1^2 > \sigma_2^2$	$\rho < 0$
	B	$\sigma_1^2 > \sigma_2^2$	$\rho > 0$
V		$\sigma_1^2 > \sigma_2^2$	$\rho = -1$

		$\bar{x}$	s^2	R
I		$\bar{x}_1 = \bar{x}_2$	$S_1^2 = S_2^2$	R = High
II		$\bar{x}_1 << \bar{x}_2$	$S_1^2 << S_2^2$	R = Medium
III		$\bar{x}_1 << \bar{x}_2$	$S_1^2 = S_2^2$	R = Light Positive
IV	A	$\bar{x}_1 = \bar{x}_2$	$S_1^2 << S_2^2$	R = High Negative
	B	$\bar{x}_1 = \bar{x}_2$	$S_1^2 << S_2^2$	R = Medium Postive
	C	$\bar{x}_1 = \bar{x}_2$	$S_1^2 << S_2^2$	R = High Positive

		σ^2	$\rho =$
I		$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} = 0$
II	A	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ Low
	B	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ High
	C	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$
III		$\sigma_1^2 > \sigma_2^2 > \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ High
IV	A	$\sigma_1^2 > \sigma_2^2 > \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$
	B	$\sigma_1^2 > \sigma_2^2 > \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$

	$\bar{x}$	s^2	R
I	$\bar{x}_1 = \bar{x}_2 = \bar{x}_3$	$S_1^2 = S_2^2 = S_3^2$	$R_{12} = R_{13} = R_{23}$ = Very High
II	$\bar{x}_1 < \bar{x}_2 < \bar{x}_3$	$S_1^2 < S_2^2 << S_3^2$	$R_{12} \neq R_{13} \neq R_{23}$ = Medium Values
III	$\bar{x}_1 < \bar{x}_2 < \bar{x}_3$	$S_1^2 < S_2^2 < S_3^2$	$R_{12} \neq R_{13} \neq R_{23}$ = High Values
IV	$\bar{x}_1 = \bar{x}_2 = \bar{x}_3$	$S_1^2 < S_2^2 < S_3^2$	R_{12} = High Negative R_{13} = Low Positive R_{23} = High Positive

When a case calls for a test of the mean vector that test takes the following form:

Null hypotheses: $\underline{\mu} = \underline{\mu}_0$ (where the student enters $\underline{\mu}_0$)

The confidence region is defined by the c value which is related to the F statistic (see Section 3.2). The null hypothesis is not rejected if the point representing the mean vector is within the confidence region and rejected if it is outside the confidence region. This is a simple but powerful visual test. The test of the mean vector will always be conducted with the normal distribution (bivariate or trivariate) characterized by $\bar{x}$ and $\frac{S}{n}$ (sample data).

Case 1 deals with the Bivariate normal distribution and deals only with theoretical data. The options for case 1 are outlined in Table 3.3.

TABLE 3.3. Case 1 Options

		σ^2	ρ
I		$\sigma_1^2 = \sigma_2^2$	$\rho = 0$
II	A	$\sigma_1^2 < \sigma_2^2$	ρ = Low
	B	$\sigma_1^2 = \sigma_2^2$	ρ = High
III	A	$\sigma_1^2 > \sigma_2^2$	ρ = Constant
	B	$\sigma_1^2 < \sigma_2^2$	ρ = Constant
IV	A	$\sigma_1^2 > \sigma_2^2$	$\rho < 0$
	B	$\sigma_1^2 > \sigma_2^2$	$\rho > 0$
V		$\sigma_1^2 > \sigma_2^2$	$\rho = -1$

Option 1: The total variance is the sum of the variances. Option 1 investigates the situation where the covariance of each pair of variables is 0 and the variable means and variances are equal. This means that knowing the value of x_2 yields no information concerning the value of x_2. This option is geometrically pictured as a perfect circle.

Option 2: Compares the differences between a case of low covariance verses a case of high covariance. The means and variances are equal in each case so that they have no effect on the comparison.

Option 3: Compares two runs. The first run has σ^2 σ_2^2 and the means and covariances equal. The second run has $\sigma_2^2 = \sigma^2$. This option is intended to show how differences in variances effect the size and shape of the geometric picture. It also initiates the students understanding of the interrelation-

ship between variance and covariance. The student should notice the elongation of the ellipse in the direction of the variable with the larger variance. Option 4 stresses how covariance effects the orientation of the ellipse by comparing runs where the variance remains constant but the covariance changes sign. The student should notice a 90 degree shift in orientation.

Option 4 This case demonstrates what occurs when the two variables are perfectly linearly dependent. In this option the covariance is equal to -1 and the ellipse collapses into a line.

Case 2 is the case of the multivariate normal distribution where theoretical and empirical (from disk) data are both displayed and a test of the mean vector is conducted. The options for this case are the same as for case 1 (see Table 3.4). What is added is that the geometric images are now in

TABLE 3.4. Case 2 Options

		σ^2	$\rho =$
I		$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} = 0$
II	A	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ Low
	B	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ High
	C	$\sigma_1^2 = \sigma_2^2 = \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$
III		$\sigma_1^2 > \sigma_2^2 > \sigma_2^2$	$\rho_{12} = \rho_{13} = \rho_{23} =$ High
IV	A	$\sigma_1^2 > \sigma_2^2 > \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$
	B	$\sigma_1^2 > \sigma_2^2 > \sigma_3^2$	$\rho_{12} \neq \rho_{13} \neq \rho_{23}$
V		$\Theta_1^2 > \Theta_2^2$	$\rho = -1$

3-D. A circle becomes a sphere and an ellipse becomes an ellipsoid. The student is also introduced to the expanded concept of covariance which deals not only with a relationship between two variable but of all the interrelationships between variable pairs. He is also shown the differences in images produced from information about the population and from sample data. The test of the mean demonstrates how easy it is to determine if a null hypothesis should be rejected or not. The student needs only to determine if $\underline{\mu_0}$ is contained within the confidence ellipsoid created from sample data.

Case 3 is the bivariate normal distribution case where empirical data is entered from the keyboard. This case allows the student to experiment with different samples. The four options used

for this case represent four possible situations which could arise from different samples (see Table 3.5).

TABLE 3.5. Case 3 Options

		$\bar{x}$	s^2	R
I		$\bar{x}_1 = \bar{x}_2$	$s_1^2 = s_2^2$	R = High
II		$\bar{x}_1 << \bar{x}_2$	$s_1^2 << s_2^2$	R = Medium
III		$\bar{x}_1 << \bar{x}_2$	$s_1^2 = s_2^2$	R = Light Positive
IV	A	$\bar{x}_1 = \bar{x}_2$	$s_1^2 << s_2^2$	R = High Negative
	B	$\bar{x}_1 = \bar{x}_2$	$s_1^2 << s_2^2$	R = Medium Positive
	C	$\bar{x}_1 = \bar{x}_2$	$s_1^2 << s_2^2$	R = High Positive

Option 1: This option is a situation where there is extreme collinearity. The student can see how the ellipse begins to collapse into a line. They also can observe how the slope of the line is related to the variance and covariance (which is captured by the eigenvalues and eigenvectors).

Option 2: The second option involves the use of a sample where the means are very different and so are the variances. The student can see how this effects the shape of the images and seems to have effects related to those from different covariances.

Option 3: The third option is similar to the second except that in this case the variances are approximately equal. Therefore, any differences in the shape of the ellipse is do to the different means and that even though the variances are equal they are not equally proportional to their respective mean.

Option 4: The last option demonstrates the differences between three runs where the means are equal, the variance of one variable is much greater than the other and three different covariances are used. This option shows the user how covariance and variance interact.

The final case, Case 4, is the Multivariate normal distribution, where empirical data is entered from the keyboard. The only differences between this case and the previous one is that the images are 3-D and there are no images based on the theoretical population displayed. The options within this case are shown in Table 3.6 on the following page.

3.3-3 Criteria. Once the program was developed and the scenarios chosen and run, there needed to be a way to determine if the learning system was effective and if a positive answer could

TABLE 3.6. Case 4 Options

	$\bar{x}$	s^2	R
I	$\bar{x}_1 = \bar{x}_2 = \bar{x}_3$	$S_1^2 = S_2^2 = S_3^2$	$R_{12} = R_{13} = R_{23}$ = Very High
II	$\bar{x}_1 < \bar{x}_2 < \bar{x}_3$	$S_1^2 < S_2^2 << S_3^2$	$R_{12} \neq R_{13} \neq R_{23}$ = Medium Values
III	$\bar{x}_1 < \bar{x}_2 < \bar{x}_3$	$S_1^2 < S_2^2 < S_3^2$	$R_{12} \neq R_{13} \neq R_{23}$ = High Values
IV	$\bar{x}_1 = \bar{x}_2 = \bar{x}_3$	$S_1^2 < S_2^2 < S_3^2$	R_{12} = High Negative R_{13} = Low Positive R_{23} = High Positive

be suggested to the investigative question posed by this thesis. The only meaningful criteria was to determine if the learning system fostered the creation of new knowledge in the area of covariance. In other words, did the computer program display the appropriate statistical values and geometric images which when used in a VVAM governed environment would stimulate the student to ask significant and relevant questions about the subject and then supply enough information to answer those questions. If the answer is yes, then the student was able to link the knowledge they brought to the learning experience with new knowledge they constructed through this question/answer cycle. The question/answer cycle should continue until the subject is exhausted because, as the student creates new knowledge it will allow them to ask new questions and get new answers (create new knowledge). It should be possible for this cycle to repeat indefinitely.

IV.Findings and Analysis

4.1 Introduction

Once the program itself was completed, tested and debugged it was then run using the cases delineated in Chapter 3. This was done to determine if it provided the necessary information to the student to stimulate meaningful and relevant questions and supply sufficient information to answer those questions thereby encouraging the creation of new knowledge in the area of covariance. The cases used are shown in Table 3.2 (see page 3.20) and summarized in Table 4.1

TABLE 4.1 Summary of Cases Run

Cases Run			
Case 1	**Case 2**	**Case 3**	**Case 4**
Bivariate Normal	Multivariate Normal	Bivariate Normal	Multivariate Normal
BVN	MVN	BVN	MVN
Theoretical	Theoretical/Empirical Data generated from computer	Empirical Data manually entered	Empirical Data from disk

Theoretical means that geometric images displayed to represent the population, either BVN or MVN, were created using the populations characterizing parameters ($\underline{\mu}$ and Σ). When a case states that it contains empirical data it means that the geometric images used to represent the population from which the sample (empirical) data were taken are based on the estimators of the characterizing parameters ($\bar{x}$ and S) of that population. Empirical data is supplied to the computer in one of three manners. First it could be generated by the computer from the population parameters. How this is done is explained in Appendix G. The second way empirical data can be supplied to the computer is by the student or the teacher saving it in a file called EMPDATA.DAT in the same directory as the program. The final method of supplying empirical data to the computer is by typing it in via the keyboard while running the program.

There were several reasons why these four cases were chosen out of the 24 possible cases. First of all the time frame available for this research was prohibitive not allowing all cases to be run. The cases chosen were done so because they not only cover most of the computer programs

capabilities, but they also cover nearly all the significant concepts of covariance which the student is expected to be able to create. The full collection of screen captures (both inputs and outputs) from the running of all four cases and their associated options is included as Appendix A. When the selected cases were run valuable information was gathered which suggested a positive answer to the research's investigative question.

4.2 Study of Case 1

The first case run was the case involving the Bivariate normal distribution and containing only theoretical data. Table 3.4 (see page 3.22) contains the details on the five options associated with this case. The output for Case 1 Option 1 is shown in Figure A.2 (see page A.2). The screen is divided into four subsections. The top left section (not labeled in the output) contains statistical values such as the mean vector, the variance/covariance matrix, the eigenvalues, eigenvectors and the $\frac{\Sigma}{n}$ matrix and its associated eigenvalues and eigenvectors. The top right section (labeled screen 1) contains an image of the contours representing the normal distribution with the parameters $\underline{\mu}$ and $\frac{\Sigma}{n}$. The major and minor axis shown with this image are those for the distribution characterized by the parameters μ and Σ. This is done to give scale to the image so it can be compared with the image in the bottom left section. This section (labeled screen 2) contains the contours of the density function characterized by the parameters $\underline{\mu}$ and Σ. The final section of the screen (labeled screen 3) is left blank for this case. This section of the screen is used in the other cases to display the test of the mean vector.

What the student observes from this first option is that the ellipses in both image sections of the screen are in fact perfect circles and the angles of rotation of these circles are both 0.0 degrees. He can also see that the major and minor axis of the image in screen 1 is scaled by the factor n from those in screen 2. It can be seen that in this option knowing one of the variables values yields no information about the other. At this point the student has successfully created knowledge on what it means for two variables to be independent. In other words, he has synthesized knowledge from a single special case. This information supports the second hypothesis that computer generated graphical images embedded in appropriately pictorialized vector spaces can be employed to enhance the identification and understanding of critical statistical features traditionally used to characterize the covariance structure of bivariate normal populations. It also provides support for

the third hypothesis that the VVAM protocol can enable a student to gain a profound and experimentally based mastery of the concepts of covariance.

The second option is divided into two separate runs for the student to compare and contrast. The output for part A of this option is Figure A.4 (see page A.2) and the output for part B is Figure A.6 (see page A.3). When the student examines these two parts they see that in part A, where the covariance is low, that the ellipses are slightly elongated and there is an equal angle of rotation for the images in both screen 1 and screen 2. In part B, where the covariance is high, the ellipses are much more elongated but the angle of rotation is the same as in part A. The student learns that the covariance is responsible for the shape of the ellipse. They also are stimulated to ask why there is now an angle and, since it remains constant when the covariance changes, what controls it. This is an example of experimental based learning. This supports the first research hypothesis that the VVAM can be used to orchestrate a experiment-based environment. The next option is in response to those unanswered questions.

The third option is also divided into two parts. In both these parts the covariance remains constant at a high value while the variances of each variable are swapped from one part to the other. The output for part A is Figure A.8 (see page A.4) and the output for part B is Figure A.10 (see page A.5). When the student compares these two outputs he sees that the ellipses are all elongated as they could have predicted from what they learned from the last option. They also see that the angles of rotation for the ellipses are not the same for both parts. Since the only difference between these runs is that the variance changes, the student can determine that it is the variances which control the angle of rotation of each ellipse. Again the student has created new knowledge from a self-discovery learning environment.

Using the information from options two and three, the student has synthesized knowledge from information gained within a case. This demonstrates that the program has been able to stimulate meaningful learning on a second level. The first being from a single special case as was demonstrated with option one.

In option 4, the student is also guided through a comparison of two runs. In both parts the relationship between the variances is the same while the covariance is high and negative for part A (Figure A.12, page A.6) and is low and positive for part B (Figure A.14, page A.7). From observing the output of part A, the student can determine that the negative covariance causes the slope of the major axis to be negative. This is confirmed when he views the output from part B and

sees that the slope is again positive which corresponds to the covariance. In both outputs the ellipses are elongated as would be expected, now that the student knows that this is caused by the presence of a non-zero covariance. There is also consistency in their observations because the greater elongation occurs in part A, where the convariance is higher.

In the final option of this case, the user is shown the effect on the images when the covariance is both negative and equal to 1. They can hypothesize that the slope of the major axis will be negative from what they learned from the last option. What they also now see is that the ellipses are no longer ellipses but have degenerated into lines. The student can determine that this is the reverse of option 1 when covariance was zero. Now only one variable needs to be known in order to find a point from the distribution. In this case of perfect (negative) covariance, one variable is perfectly predictable from the other. Again the student has synthesized knowledge from two levels, both a single special option, perfect covariance, and from the comparison of two options within a single case, no versus perfect covariance.

With the knowledge the student has created from this first case, he can view the second case with certain hypotheses going in. He is in fact creating an experiment by which he can test those hypotheses. This is exactly the type of behavior which the three research hypotheses suggest would occur if a graphics oriented program, like the one developed here, is used within the VVAM protocol. This is strong support for the investigative question.

4.3 Study of Case 2

Case 2 involves the Multivariate normal distribution where geometric images for both theoretical and empirical data are displayed. This case also contains a test of the mean vector for each of the options which appears in screen 3. See chapter III for a review of the testing procedure. Table 3.4, see page 3.22, contains all the options which make up this case. Since the options will result in displays of both the theoretical and empirical distributions the student will be able to readily compare and contrast them.

The first option is the special case on independence as was seen in the first option of Case 1. The student is able to recognize that the pairwise covariances are all zero and from this he can then hypothesize that the images representing the distribution in any of the planes defined by two of the three variables would be a circle. Therefore, the image representing the MVN, all three variables simultaneously, would be a sphere. The output for option one is Figure A.9 (see page A.5). It becomes readily apparent that the users hypothesis is correct, that the theoretical images are indeed

spheres. However, he quickly realizes that the images for empirical data are not spheres but ellipsoids. Using what was learned from the first case, he can see that this is not totally unexpected since the data on the input screen of this option (Figure A.17, page A.9) shows that the pairwise covariances for the sample data are not equal to zero. What the student learns from this is that sample data from a population may not have the same exact characteristics as the population. This is the first step in understanding the idea of estimators, a vital concept in statistics. Screen 3 contains the test of the mean vector. If the student focuses his attention on this screen (Figure A.21, page A.11) he can see that the point representing the mean vector is within the region defined by the ellipsoid and therefore, the null hypothesis is not rejected. From this single option the student is able to learn in an experimentally based manner that:

a) the only difference between the bivariate normal distribution (BVD) and the multivariate normal distribution (MVN) is that the BVN is represented by the geometric image of a ellipse (or circle) and the MVN is represented by an ellipsoid (or sphere);

b) that the parameters calculated from a sample are only estimators of the parameters representing the population that the sample is derived from; and

c) that a test of the mean vector is simply a matter of determining if the mean vector purposed as the null hypothesis is contained within a specified confidence region.

These results are even more support for the hypothesis that this type of computer program when used with the VVAM protocol is a self discovery based learning environment which will foster the creation of new knowledge in the area of covariance.

The second option is divided into three parts. All the parts use the same variances for each variable. The first part (part A) uses the same low value for each pairwise covariance. Part B uses the same high value for each pairwise covariance. The final part uses a different high value for each of the pairwise covariances. This option parallels option 2 for Case 1. The output for part A is Figure A.23 (see page A.12). The output for part B is Figure A.27 (see page A.14) and the output for part C is Figure A.31 (see page A.16). The student hypothesizes correctly that the images from the theoretical data will differ from those of the empirical data. The student also can observe that the elongation of the ellipsoids is greater in part B then in part A, where the pairwise covariances are greater. This is consistent with what was learned in the first Case. The student can see that in all three parts the test of the mean vector results in the rejection of the null hypothesis (see Figure A.25, page A.13, Figure A.29, page A.15, and Figure A.32, page A.16, respectively for the individual tests). This option gives the user an opportunity to reaffirm what they have discovered

in the other options. This reaffirmation strengthens the students understanding by acting as positive feedback. For each assertion he makes the student can see that it is true. This gives him confidence that the new knowledge he has constructed is valid. The validation of assertions is an important component in experimental-based learning. The fact that this process is available to the learner strongly implies that indeed a experimentally-based learning environment has been constructed.

With renewed confidence in what they have already learned the student can progress to option 3. In this option the pairwise covariances are all set to the same high value. However, for this option the variances, each variables covariance with itself, are set at increasing values (i.e. $\sigma_1^2 > \sigma_2^2 > \sigma_3$). What the student can see from this option is how each pair of variables interact with each other and how all of these interactions form the ellipsoids of the distributions they are representing. The output for this option appears in Figure A.36 (see page A.18). This particular figure is a closeup on screen 3. What the user will first notice is that the mean vector representing the null is not within the region defined by the ellipsoid and therefore, the null hypothesis is rejected. He will also see, and most likely have predicted, that the ellipses for each pairwise interrelationship are at various angles to each other and to those displayed in the previous option. Looking at these more complex interrelationships leads him to ask about what happens when none of the values, variances or covariances, are kept constant. This question can be answered with option 4.

Option 4 contains two parts. Each of these parts contains a run where the variances are not equal and the pairwise covariances are also not equal (either in sign or magnitude). The output for part A of this option is Figure A.19 (see page A.10) and part B is Figure A.41 (see page A.21). By viewing these two options the student is able to start generalizing what they have seen and learned. This is an important step in the VVAM protocol because it represents the movement into the algorithmization phase. This is solid evidence that the investigative question is being positively answered. It shows that the VVAM protocol and the computer program are working together to foster meaningful learning of this complex subject area. Figure A.39 (see page A.20) shows screen 3 from part A. The student can see that the null hypothesis is supported because the mean vector is within the region defined by the confidence region. On the other hand, Figure A.42 (see page A.21) shows that the null is rejected since the mean vector is outside the confidence region. Another lesson the student will learn from this option is that they must be careful when conducting the test of the mean. They must remember that they are in three space and that the image must be looked at from several different angles to insure that the mean vector is outside the confidence region. In

Figure A.41 (see page A.21) the student may look at screen 3 and assume that the null is true but when he rotates the image as is done in Figure A.42 (see page A.21) he can see that the mean vector was actually far behind the ellipsoid. This demonstrates the power of using computer graphics to display geometric images. If the student saw the unrotated image on a blackboard he would have difficulty in understanding why the test failed when the mean vector appears to be within the ellipsoid. This apparent contradiction would disrupt the learning process. At this point, remembering back to the first case, the user may ask what would happen if all the pairwise covariances where equal to one. The fact that the student is prompted to ask this type of experimental question alone shows that he is an active part of the learning process. That he is on a voyage of self discovery learning and therefore will achieve some level of meaningful learning. This type of question also shows that the program has caused the synthesis of knowledge between cases which is learning on a third level and is strong evidence that the program and the VVAM protocol are promoting meaningful learning.

4.4 Study of Case 3.

This case involves the study of the BVN with empirical data supplied by the student from the keyboard. Before going into detail about the options of this case it is important to discuss the transition from theoretical to empirical data. In Case 1 all images were developed from purely theoretical data. In Case 2 theoretical and empirical data were given together in order to show the student how sample data is only representative of the population from which the sample came. Figure 4.1 shows the relationship between theory and practice. When information about the population is known it can be used to make deductions about a sample. When a sample is given it can be used to make inferences about the population. In the last two Cases the student will be

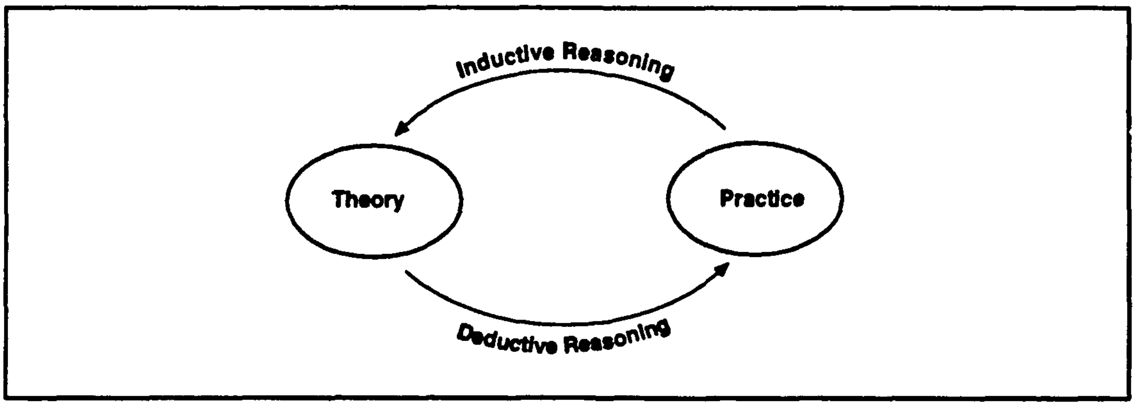


Figure 4.1. Relationship Between Theory and Practice

supplying empirical (sample) data to the program. From this data he will be able to make inferences about the population. He will also realize that these inferences take the form of estimates of the true parameters of the population. He knows this because they were able to create this knowledge while reviewing Case 2. The options for this case are described in Table 3.5, page 3.23. It should be noted that for the first time the relationships of the individual variable means are stated. This adds a new dimension to the problem. It will be the users job to determine how this information and these interrelationships between the means effect the geometric images presented on the screen.

The first option is the condition of extreme collinearity. The output for this option is Figure A.44 (see page A.22). As the student might expect when he sees that the covariance is near 1.0, the ellipses are degenerating into lines. The knowledge created from this example deals with what the characteristics of a sample are which cause extreme collinearity. He can hypothesize that since one variable is predictable when the other is present, learned from Case 1, then the predicted variable must be a scalar multiple of the predictor variable. When he reviews the input data (Figure A.43, page A.22) he will see that a given value for x_1 is nearly the same as a given value of x_2, or that $x_1 = k \times x_2$ where $k = 1.0$.

In the next option, The student supplies a sample where the means of those samples are very different and so are the variances. Examining the output (Figure A.47, see page A.24) of this option shows the student that the elongation of the ellipses is more than expected from the medium value of the covariance. Since the individual means and the variances are different the student cannot determine which caused the extra elongation. The student can explain the rotation angle of nearly 90 degrees from the extreme difference in the variances. He already constructed the knowledge that variance controls the angle of rotation form previous cases. To answer the question posed by this option the student moves on to the next option.

In this option (option 3) the hypothesis is that it is the differences in the means which has caused the extra elongation of the ellipses. When the output is studied (Figure A.51, see page A.26), the student sees that the hypothesis is in fact rejected. The ellipse is not even as elongated as the previous option and there is even a greater covariance in this option then the last. Therefore, the new hypothesis is that the increased elongation is due to the extreme differences in the variances. To test this hypothesis, option 4 is run.

The final option is used to test the new hypothesis. The means are relatively equal and the

extreme differences in variances are tested at three levels of covariance. The outputs for parts A, B and C are Figures A.53 (page A.27), A.56 (see page A.28) and A.59 (see page A.30). Close examination of these outputs suggests that the new hypothesis is not rejected. The student has modified his previous knowledge about the shaping of the ellipses with the new knowledge that differences in the variances has some effect on their shapes. This effect must be to some lesser degree than that of the convariance since it becomes apparent only at extreme differences in the variances.

What case three has shown is that the student can use this computer-based, VVAM protocol based learning environment as an experimental learning environment which is exactly what was proposed by the research hypotheses.

4.5 Study of Case 4

The description of this case is shown in Table 3.6, page 3.24. This case is the same as Case 3 except that it deals with the MVN. This case was chosen to show how the same experimentation can occur for the MVN case. The output screen for option 1 is Figure A.62 (see page A.31). This option shows the student extreme collinearity in 3-dimensions answering the question posed at the end of Case 2. The output for option 2 is Figure A.64 (see page A.32). With a closeup of screen 2 in Figure A.65 (see page A.33). The third option is displayed in Figure A.67 (see page A.34) and the final option output is Figure A.69 (see page A.35).

4.6 Summary

The findings and analysis of the methodology run has yielded strong support for the research hypotheses. The program has demonstrated its ability to facilitate the visualization and verbalization steps of the VVAM protocol. It has stimulated the creation of new knowledge on three different levels (see Figure 4.2). First it has synergized new knowledge from information, values and images,

Knowledge Creation From

Level 1→ *Visualization of Special Situations*

Level 2→ *Comparision of Options with a Case*

Level 3→ *Comparision of Different Cases*

Figure 4.2. Levels of Knowledge Creation

from special statistical situations. Second it has stimulated the creation of knowledge from the comparison of options within chosen cases of study. Finally the program was successful in prompting meaningful learning from the comparison of two or more specific statistical cases.

V. Conclusions and Recommendations

5.1 Conclusions

Previous chapters have documented the effort required to develop a learning system to display geometric images characterizing the dependent relationship among two or three multivariate normally distributed random variables. The computer program that was built to produce these displays was exercised under the governance of the VVAM protocol to answer the investigative question of this thesis, mainly: can a personal computer-based interactive graphics system be developed and employed under the aegis of the VVAM protocol to create an imaginative, visually interactive and meaningful learning environment for the study of covariance?

To answer this question, three research hypotheses had to be evaluated. This was accomplished by exercising the learning system that was developed during the thesis effort to determine if the three hypotheses could be confirmed.

Recall that the first hypothesis suggested that: the VVAM learning protocol, developed by Hansard, can be used to orchestrate an experiment-based environment for constructively learning and actively exploring variations of the covariance structures associated with selected multivariate databases.

Results of experiments reported in Chapter 4 strongly suggest that the VVAM is effective in orchestrating an experiment-based environment. Confirmation was obtained that the first two steps of the VVAM: visualization and verbalization, can be meaningfully operationalized by generating interactive displays of covariance structures. By observing and attempting to verbalize different features of ellipsoids serving to model a specific state of covariance a student is able to formulate statistical hypotheses and to design and to conduct experiments to test such hypotheses. It was also demonstrated these results could be obtained for covariance structures representing bivariate and/or multivariate distributions.

The second hypothesis suggested: computer-generated graphic images, embedded in an appropriately pictorialized vector space, can be employed to enhance the identification and understanding of critical statistical features traditionally used to characterize the covariance structure of bivariate and multivariate normal populations.

Experiments conducted to evaluate this hypothesis provided solid evidence that the graphics program produced by this thesis does, in fact, create and display images that efficaciously represent

all types of covariance structures for bivariate and multivariate normal populations. Solid confirmation was also obtained suggesting such images can be presented to the student in many alternative ways allowing him to idiosyncratically identify and master the probabilistic and statistical theory associated with particular covariance structures. More importantly, such pictures encourage him to do this in an entirely intuitive manner under severe time constraints. This ensures the learning system can find wide application in the conventional classroom.

Finally, note that the third hypothesis proposed:

Exercising a graphics oriented program which supports first two steps of the VVAM protocol: visualization and verbalization and which implements relevant heuristics for constructive learning will enable students to gain a profound and experiment-based mastery of the multivariate covariance analysis concepts.

In fact, as soon as the program was run under the aegis of the VVAM it became obvious that any student will be greatly assisted in his efforts to learn by discovery by such an image rich environment. Interactive visual aids help foster an awareness of subtle connections among variables and facilitate competent articulation of the explicit and implicit principles of multivariate statistics that are mirrored by the size, shape and orientation of an elliptical image of covariance. Experiments run and reported in Chapter IV confirmed that a student, under competent instructor guidance can, and will, use his personally constructed knowledge to link whatever he knows coming into the learning experience to what he is able to actively learn in the environment facilitated by the graphical display system of this thesis.

Students routinely formulated questions during the time they were verbalizing about what they saw on the computer screen. Such spontaneous behavior confirmed the graphics program produced by this thesis does facilitate an environment that stimulates questioning and encourages a student to attempt to answer his own questions. Immediate and continuous feedback guarantees the student will maintain a lively interest in the subject under study as well as experience profound enthusiasm for carrying out whatever learning activities he is requested to accomplish.

In summary, abundant evidence was obtained to suggest full and complete support exists for each of the research hypotheses. Thus, a positive response to the investigative question can be made; that is, that a personal computer-based interactive graphics system can be, and now has been, developed and implemented under the aegis of the VVAM protocol to create an imaginative, visually interactive and meaningful learning environment for the study of covariance.

5.2 Recommendations for Future Research

Three suggestions for future research are

1. To enhance the current computer program.
2. To generate displays that display the evolution of Covariance Structures over some continuum like time, space or the domain of one or more population parameters
3. To adapt the VVAM protocol for use in a qualitative context

5.2-1 Enhancing the Current Program.

Several specific enhancements to the current program could expand a student's ability to construct new knowledge of covariance structures. First, if the program were restructured to handle larger sample sizes more extensive explorations could be made concerning the sensitivity of covariance structures to modifications in population parameter values. This would enhance a student's ability to explore the relationship sample size has on attempts to estimate or test hypotheses about a populations characteristics.

Enhancements would also permit the study of a two samples scenario. This would require creating and displaying images of two covariance structures simultaneously in order to permit the comparison of two multivariate normal populations. Such an enhancement would allow a student to explore concepts of two-sample inference as well as open up possibilities for studying additional multivariate topics such as Multivariate Regression, Discriminate Analysis and Multidimensional Scaling.

5.2-2 Generation of Evolving Displays of Covariance.

Another recommendation would be to take all the current and proposed capabilities of the program and create a new computer program to facilitate the animated display of evolving ellipsoids - each subject to change and variation through time. This new capability could be considered as an attempt to display the evolution of covariance as it changes through time - which it surely does in the real world. It would also require a very computationally intense environment and pose extraordinary demands on internal memory. A Sun work station would support such fast computation needs as well as execute Mathematica: software that already contains relevant mathematical and animation routines as part of its menu of callable procedures.

Since the current program requires students to visualize ellipsoids portrayed by three defining ellipses, sometimes it is difficult to maintain a full 3-D perspective, especially if two ellipsoids are displayed at the same time. If each ellipsoid was presented as a hollow shell the student could easily

distinguish between separate ellipsoids on the screen. Animation coupled with such a shell image would facilitate dynamic and coherent displays of relevant ellipsoids.

When the current program is run and a student wants to change one or more of the values he entered as input, he must restart the program and initiate a new run of the program. Full animation would allow him to change values periodically (e.g., one or more of the parameters in the mean vector or covariance matrix) and obtain instant feedback on how such modifications effect the shape, size or orientation of the ellipsoid. This would heighten the level of active exploration and experimentation and further enhance a student's opportunity to discover his innate ability to construct new knowledge at any level of abstraction.

5.2-3 Adaptation of the VVAM Protocol for Qualitative Contexts. This final recommendation promises to yield the most profound enhancements of all. For if the VVAM protocol is applied in a non-mathematically based curriculum, construction of new knowledge can occur in any conceivable learning environment. In fact, this research demonstrated again and again that the most important steps of the VVAM protocol are the first two; visualization and verbalization. These two steps should remain relatively unchanged in a qualitative environment. The difficult part would be finding an appropriate way to logically algorithmitize and formally articulate the new knowledge acquired. Visualizations could be facilitated, perhaps, by an active involvement with computer based videos, and by orchestrating role playing exercises or computer based simulations. It is important to emphasize that whenever the VVAM is exercised in a qualitative context the student should always be allowed to maintain active control over what is being visualized. He should also be able to experiment with self-selected situations and formulate and test hypotheses he considers important. The use of heuristics rather than algorithms would probably serve as step three of the VVAM protocol. In lieu of using the language of mathematics for formalization at Step 4, a more ideographic language could be employed offering much richer variety for expressing any given heuristic. This would surely be mandatory during the study of living systems,.

In short, the research performed to complete this thesis confirmed that the personal computer can be used to orchestrate a Socratic dialogue among the three major components of the learning triad:

Student-Curriculum-Teacher

within the constraints of conventional school governance. It also confirmed this can be done in

a way that facilitates an environment for meaningful learning and stimulates self-discovery of the concepts and propositions of multivariate covariance analysis.

Appendix A

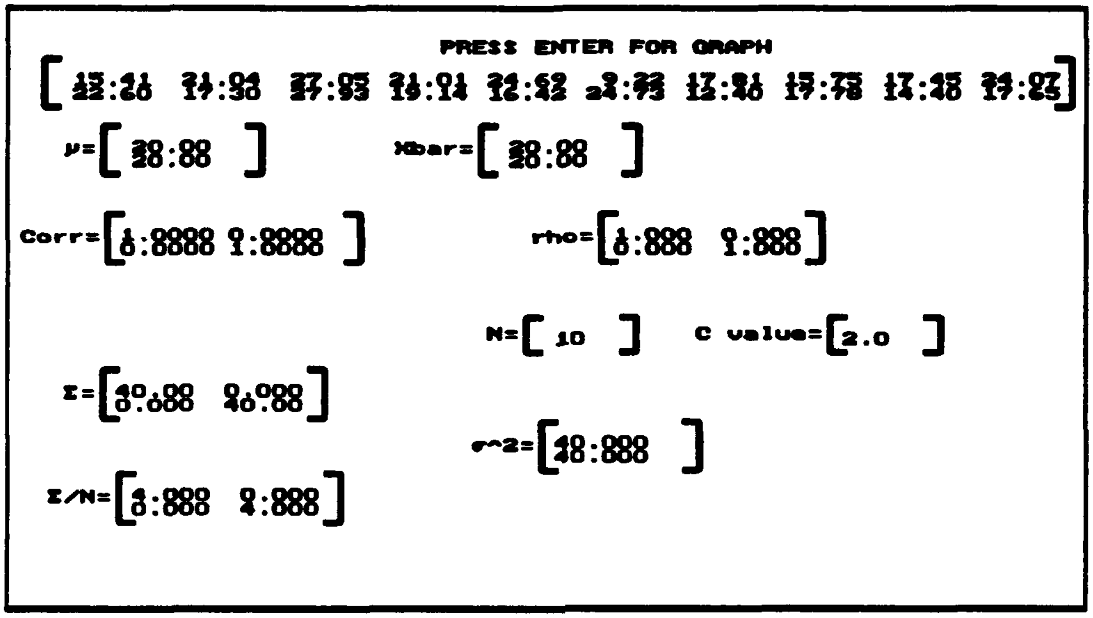


Figure A.1 Input Screen for Case 1 Option 1: Independence

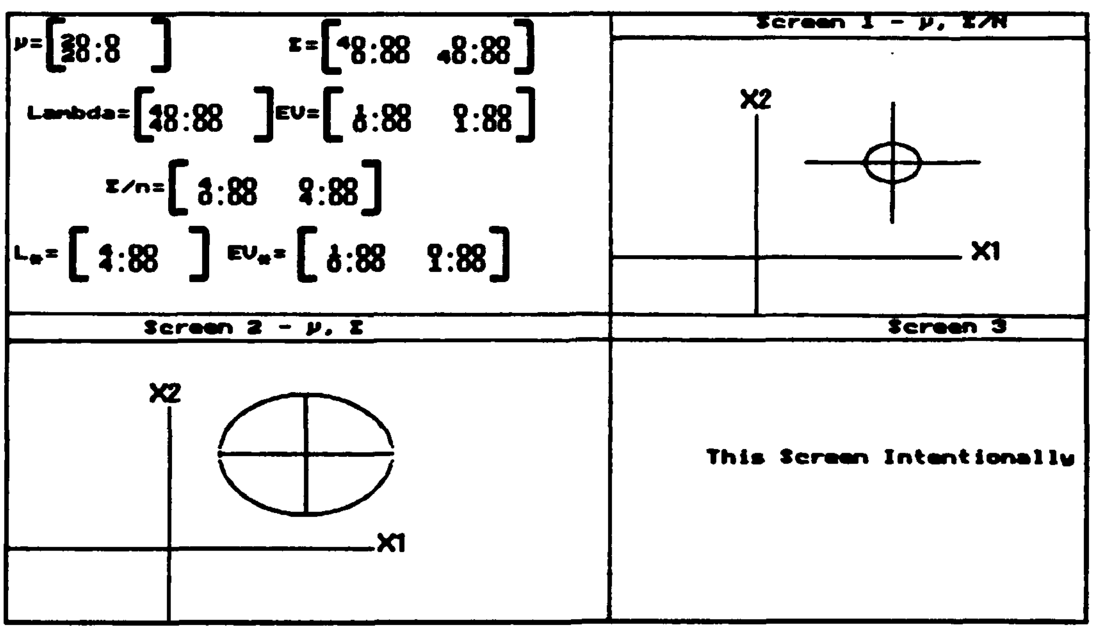


Figure A.2 Output for Case 1 Option 1

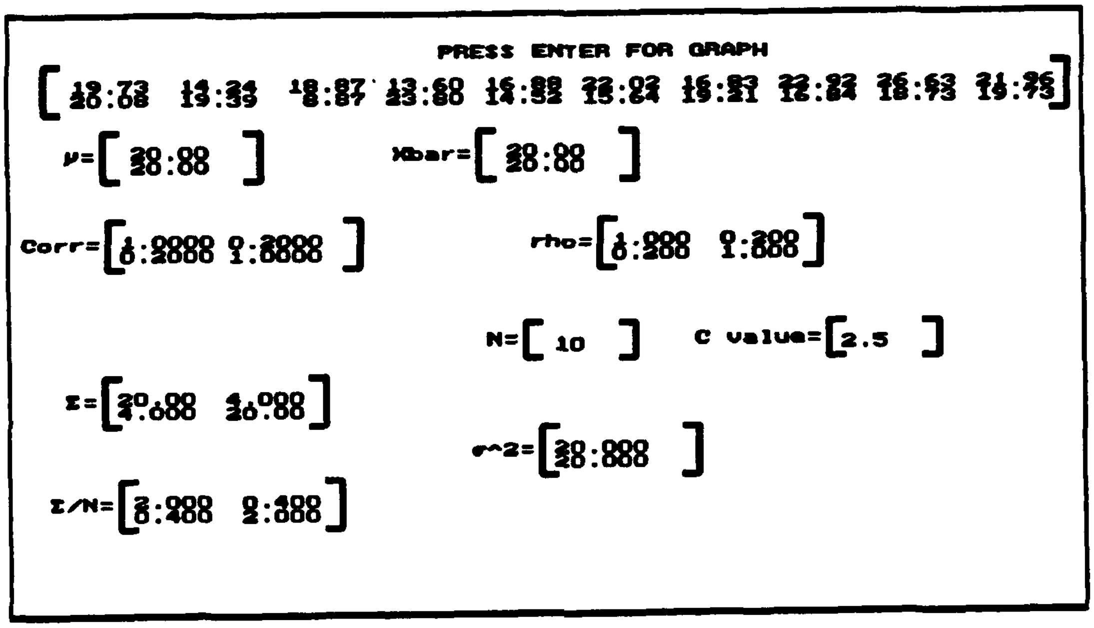


Figure A.3 Input Screen for Case 1 Option 2A

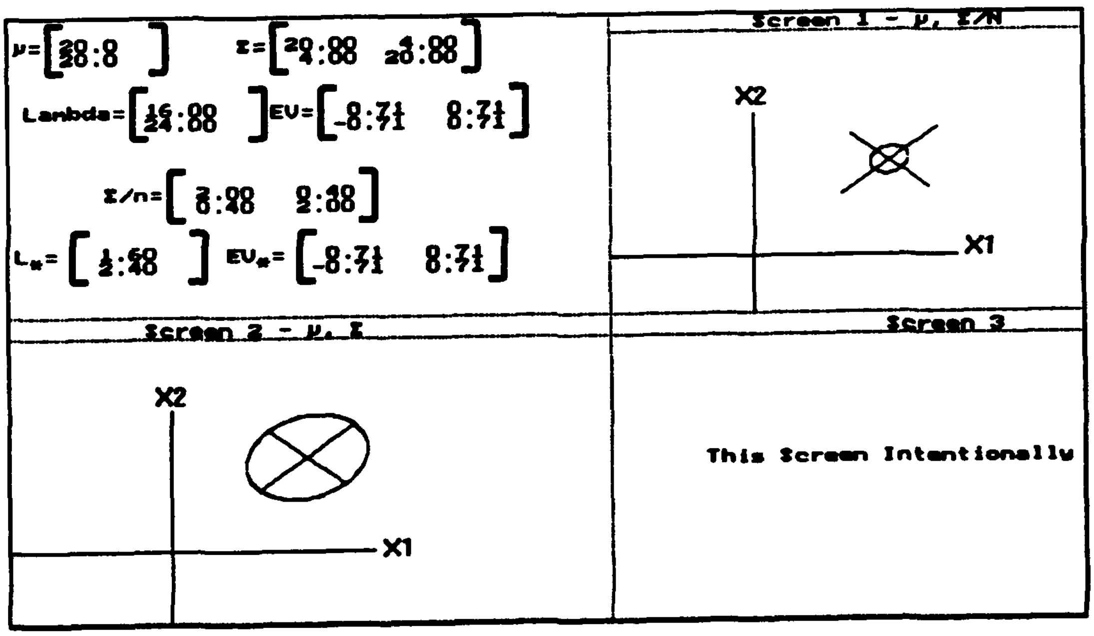


Figure A.4 Output Screen for Case 1 Option 2A

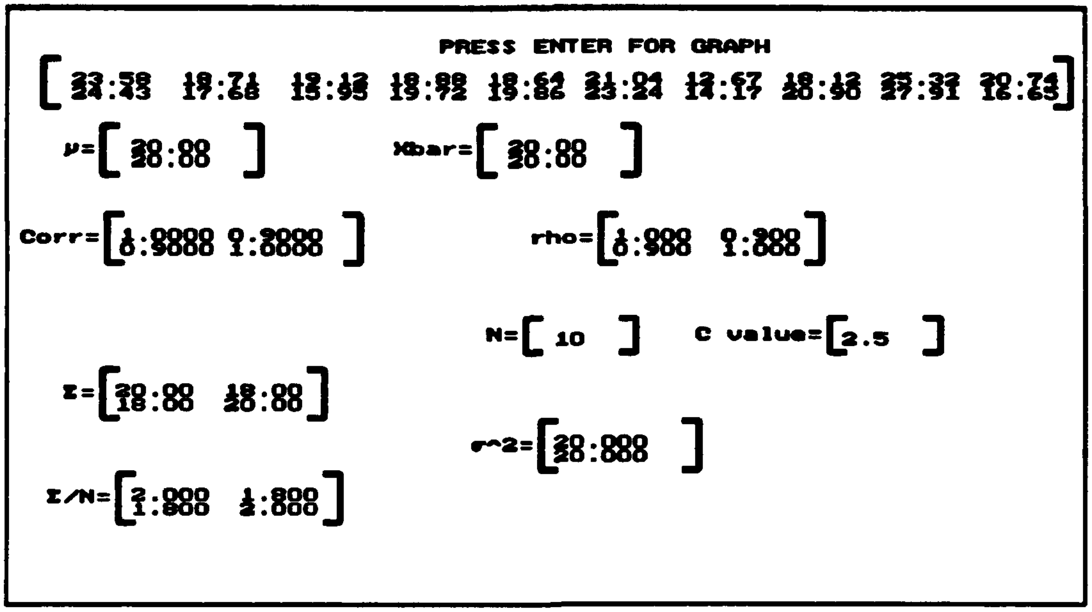


Figure A.5 Input Screen for Case 1 Option 2B

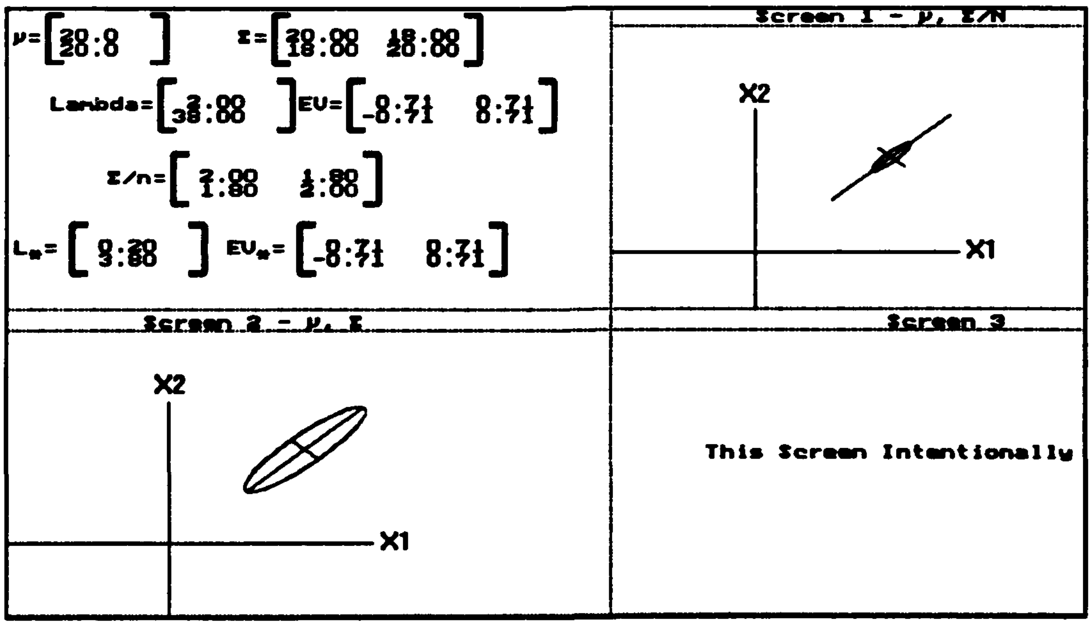


Figure A.6 Output for Case 1 Option 2B

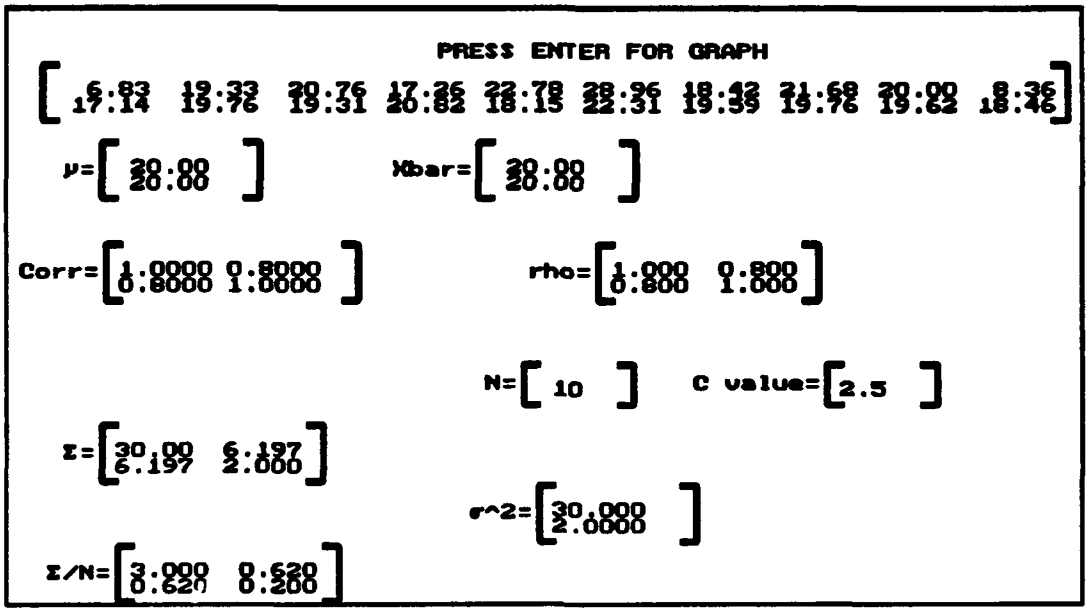


Figure A.7 Input Screen for Case 1 Option 3A

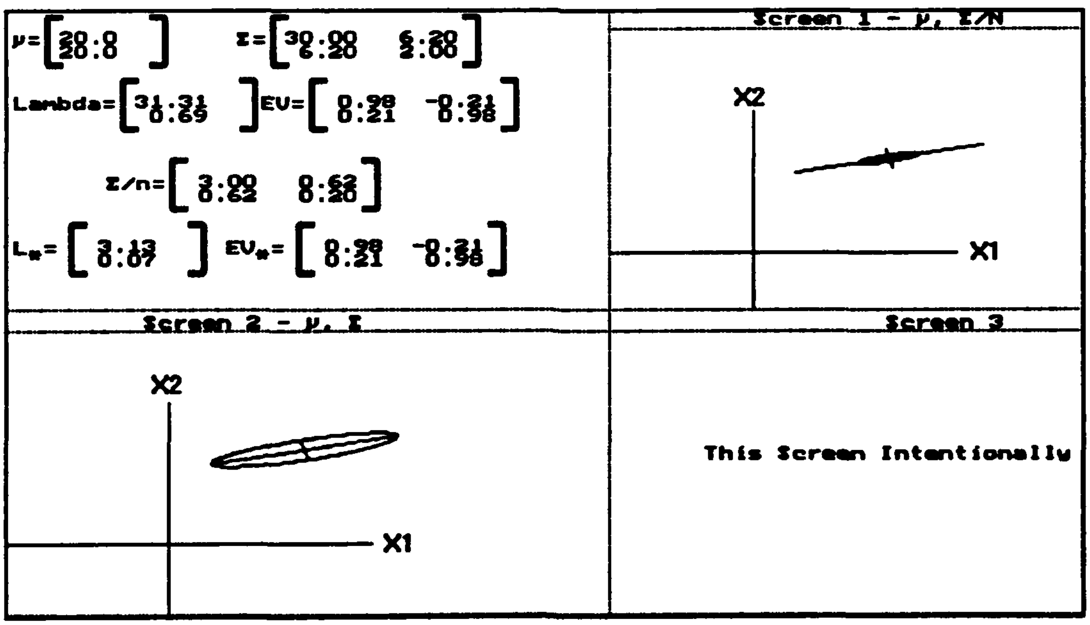


Figure A.8 Output for Case 1 Option 3A

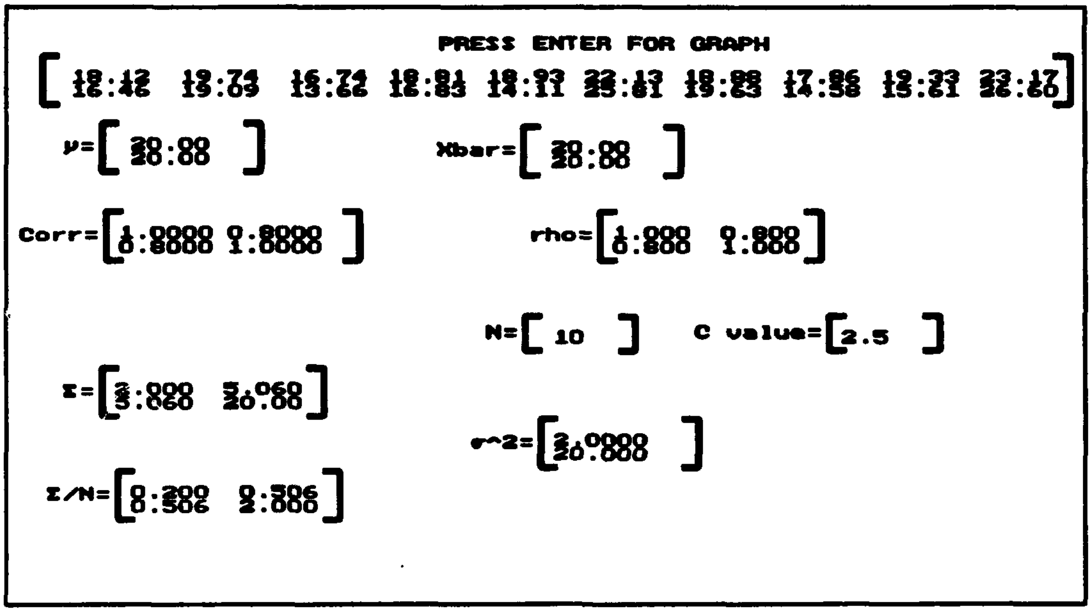


Figure A.9 Input Screen for Case 1 Option 3B

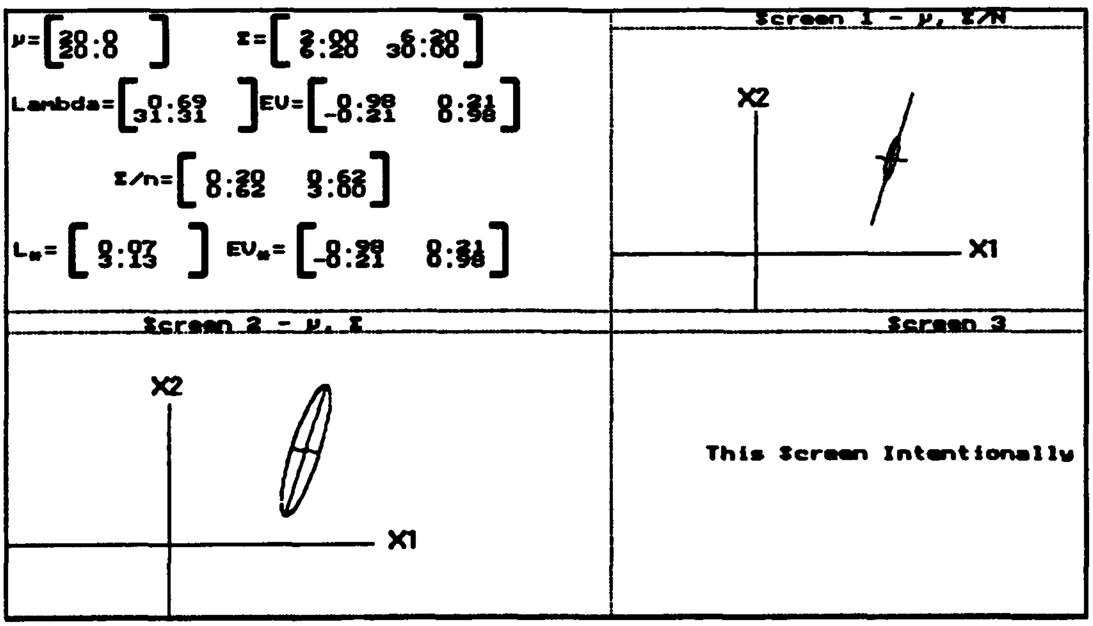


Figure A.10 Output for Case 1 Option 3B

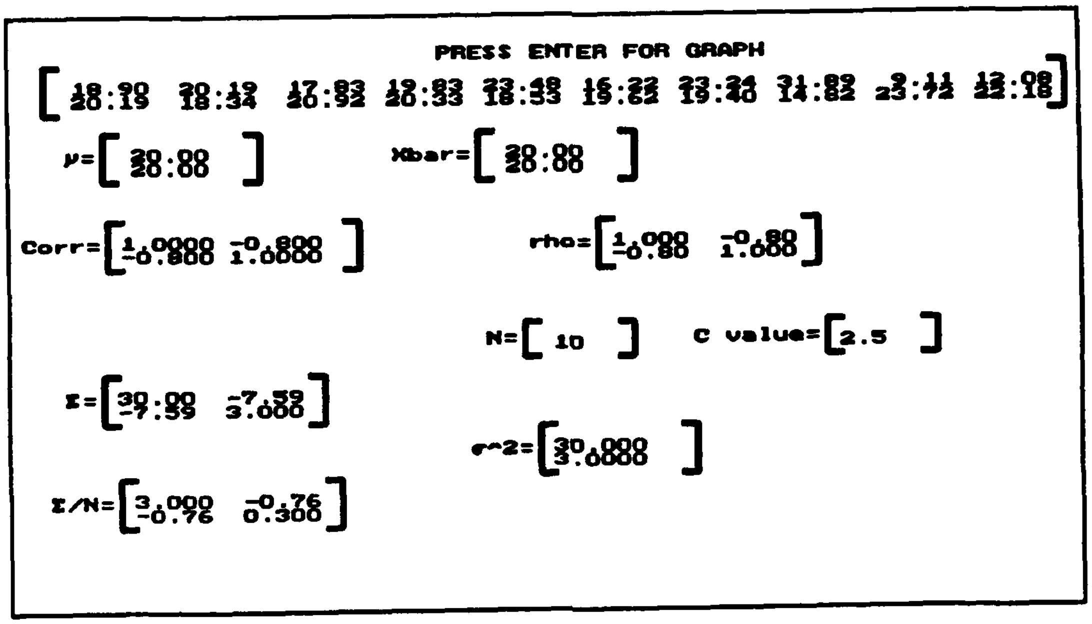


Figure A.11 Input Screen for Case 1 Option 4

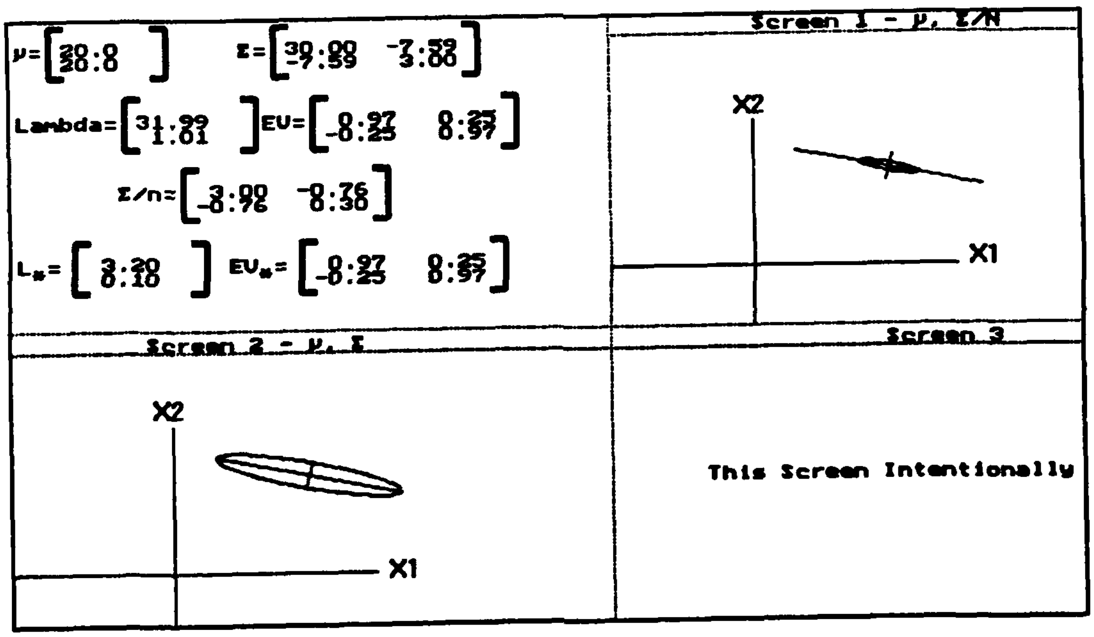


Figure A.12 Output for Case 1 Option 4A

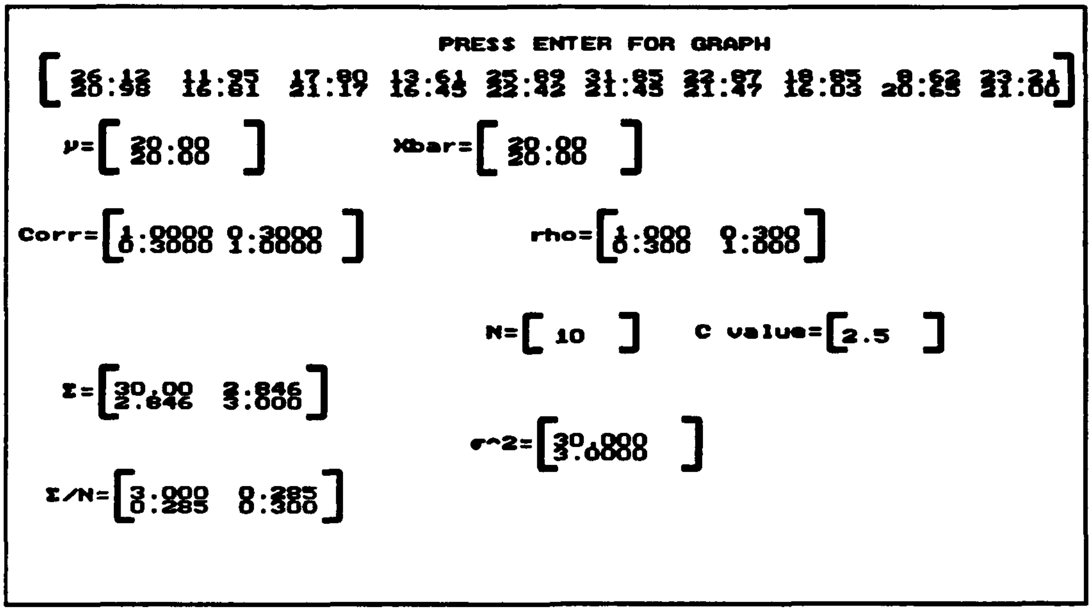


Figure A.13 Input Screen for Case 1 Option 4B

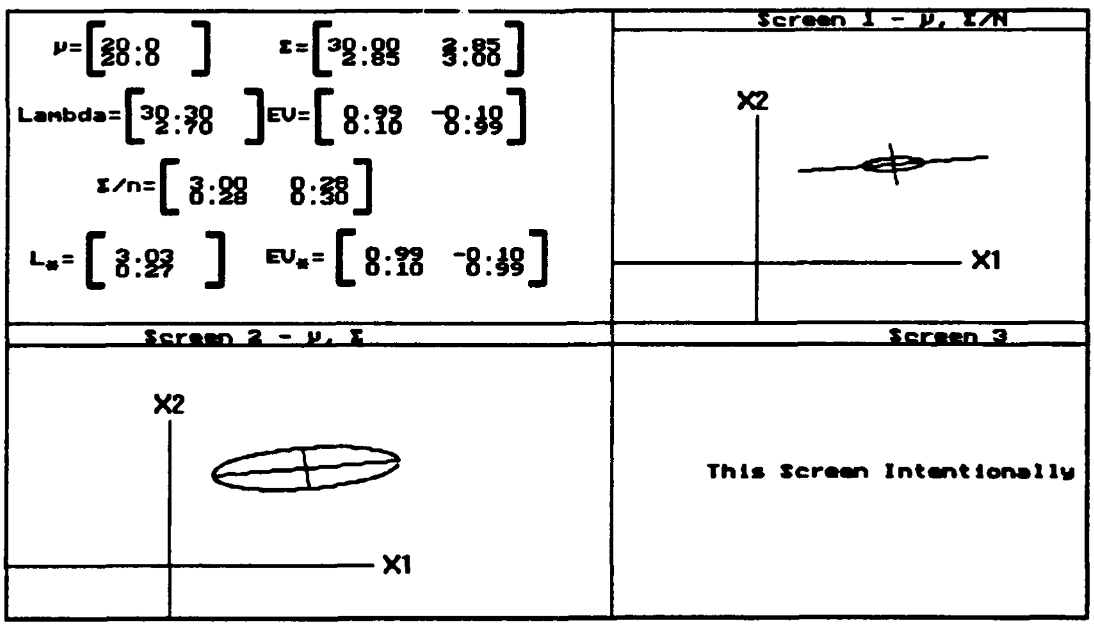


Figure A.14 Output for Case 1 Option 4B

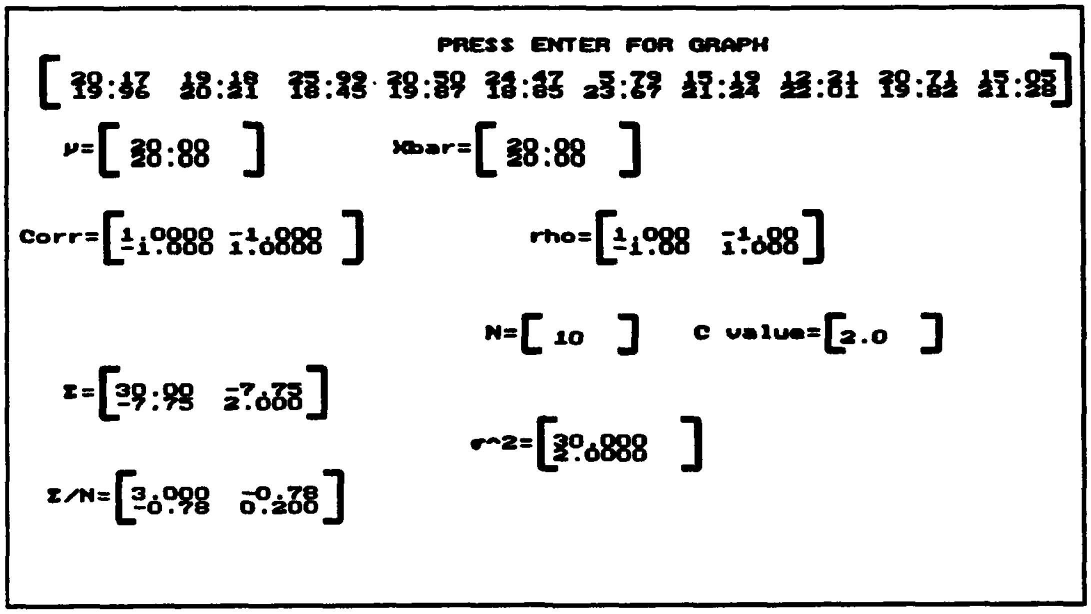


Figure A.15 Input Screen for Case 1 Option 5

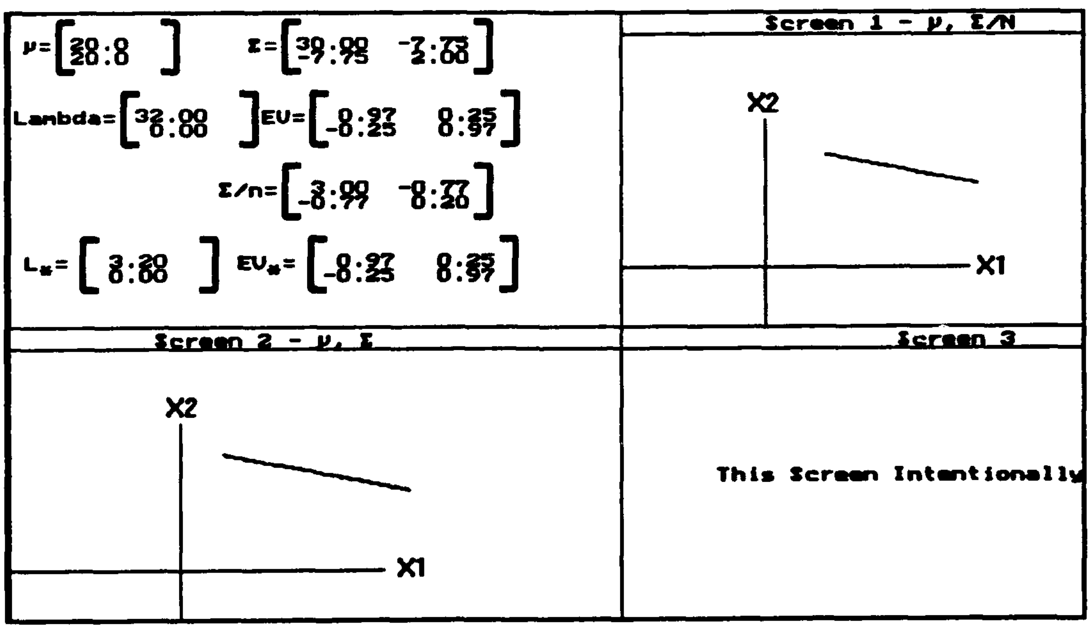


Figure A.16 Output for Case 1 Option 5

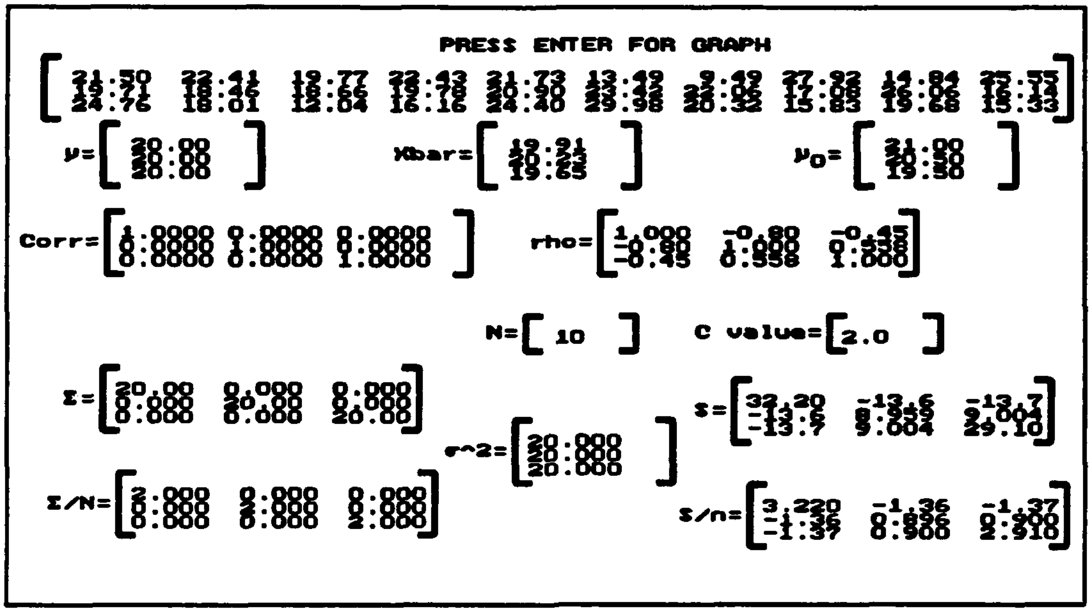


Figure A.17 Input Screen for Case 2 Option 1

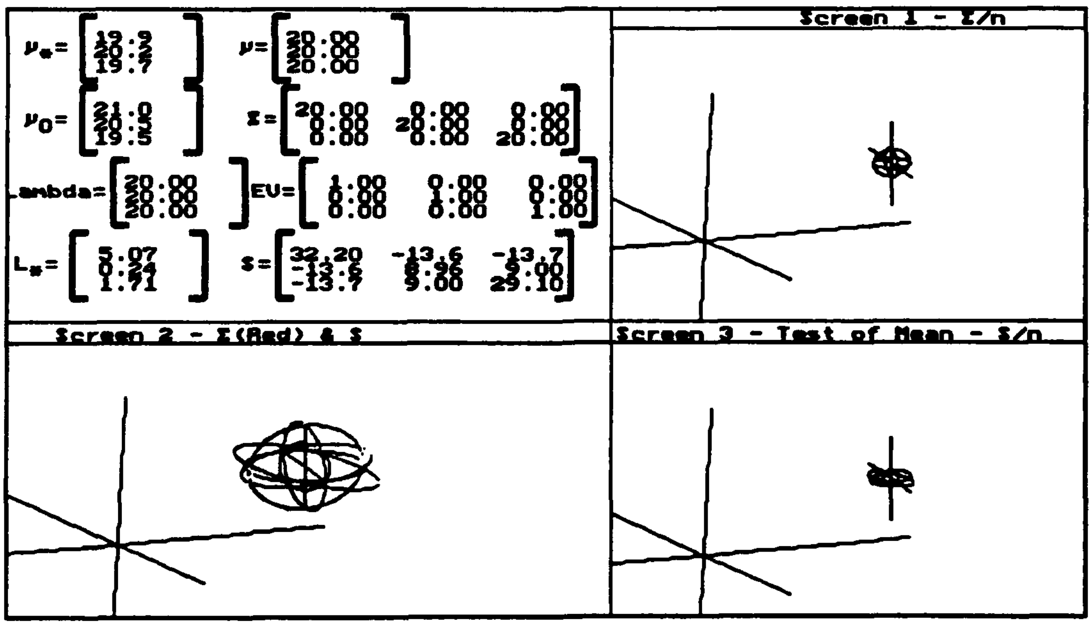


Figure A.18 Output for Case 2 Option 1

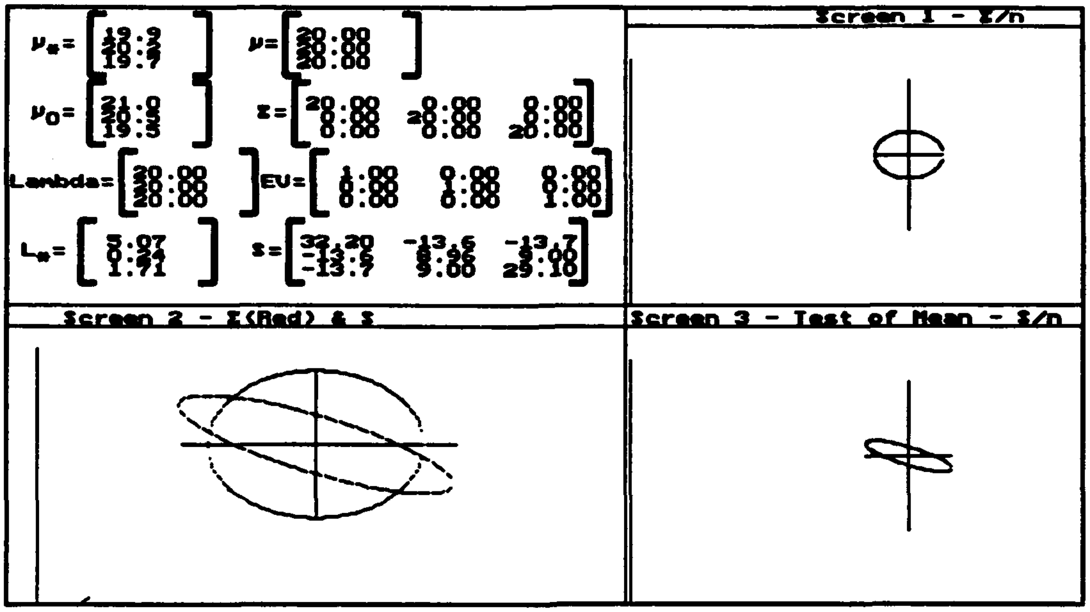


Figure A.19 Output Screen for Case 2 Enlarged View

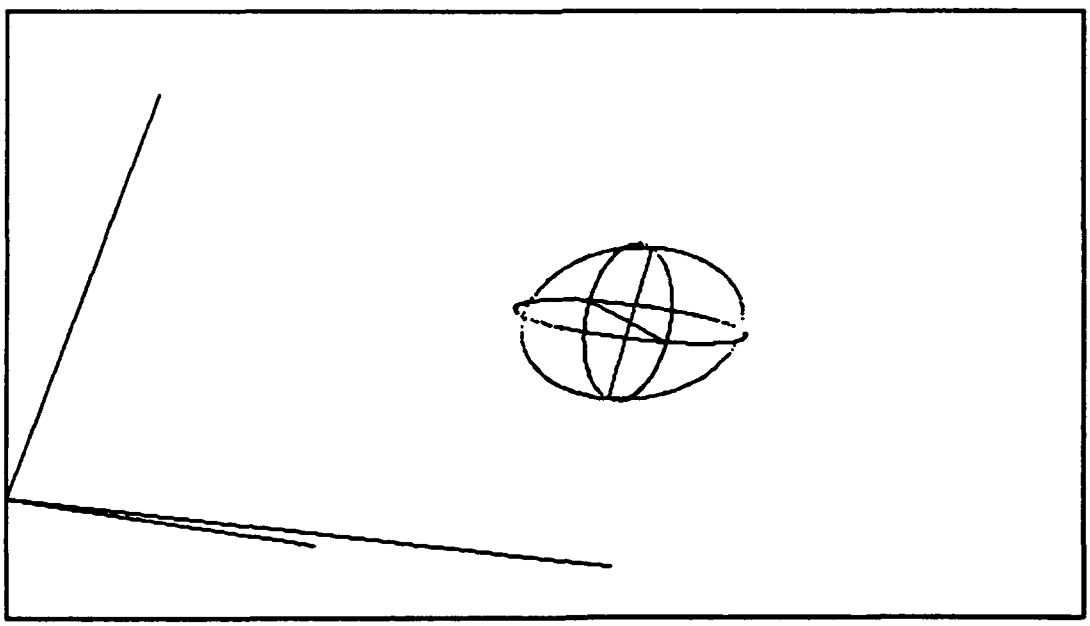


Figure A.20 Output for Case 2 Focus on Screen 2

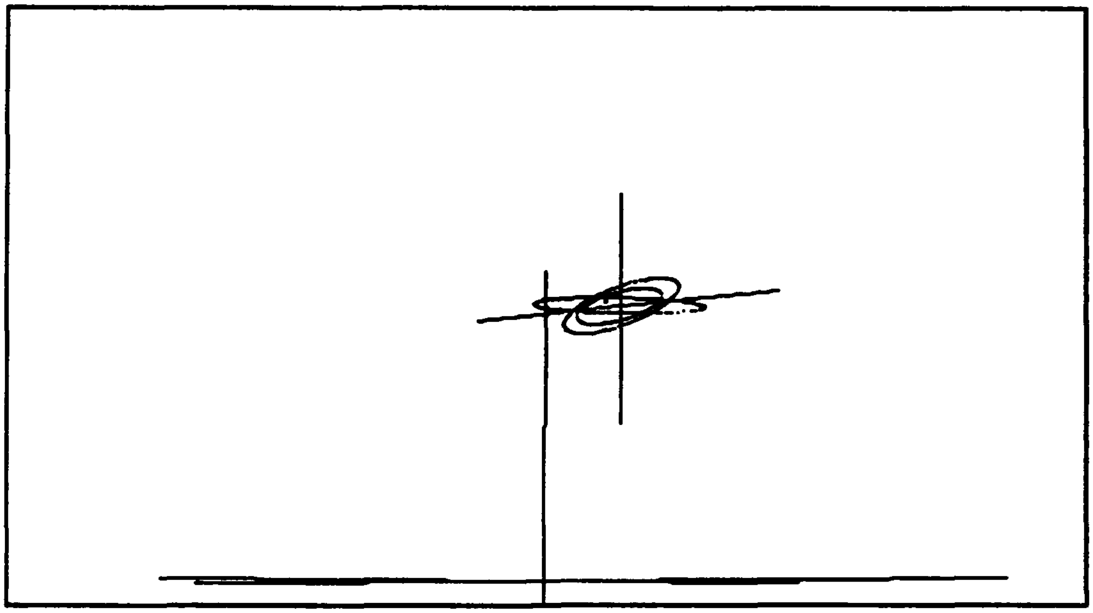


Figure A.21 Output Case 2 Option 1 Focus on Screen 3

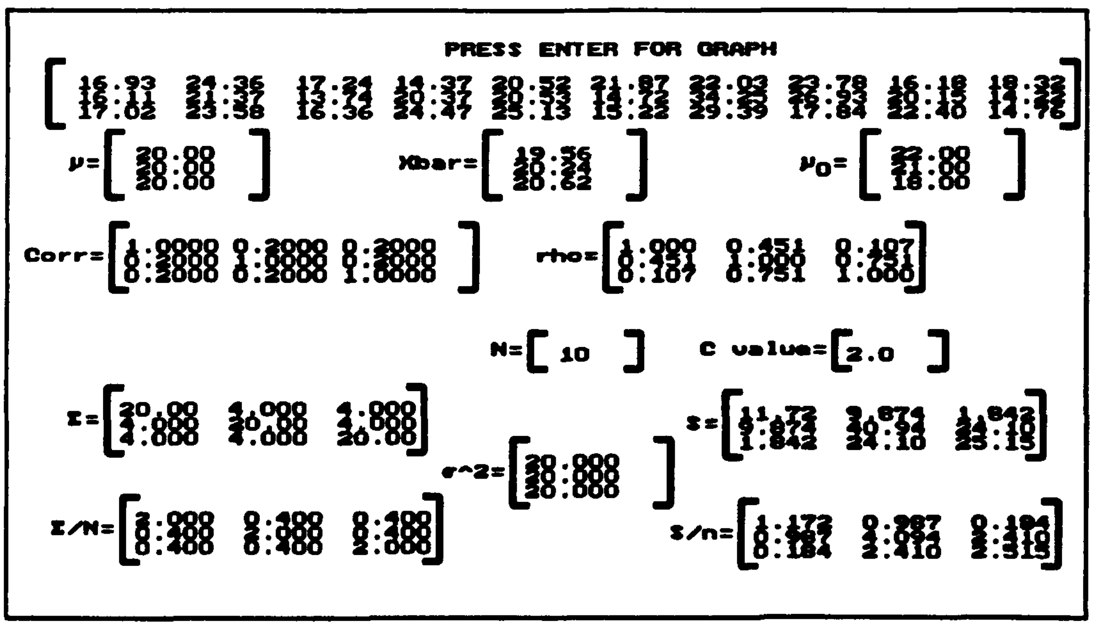


Figure A.22 Input Screen for Case 2 Option 2A

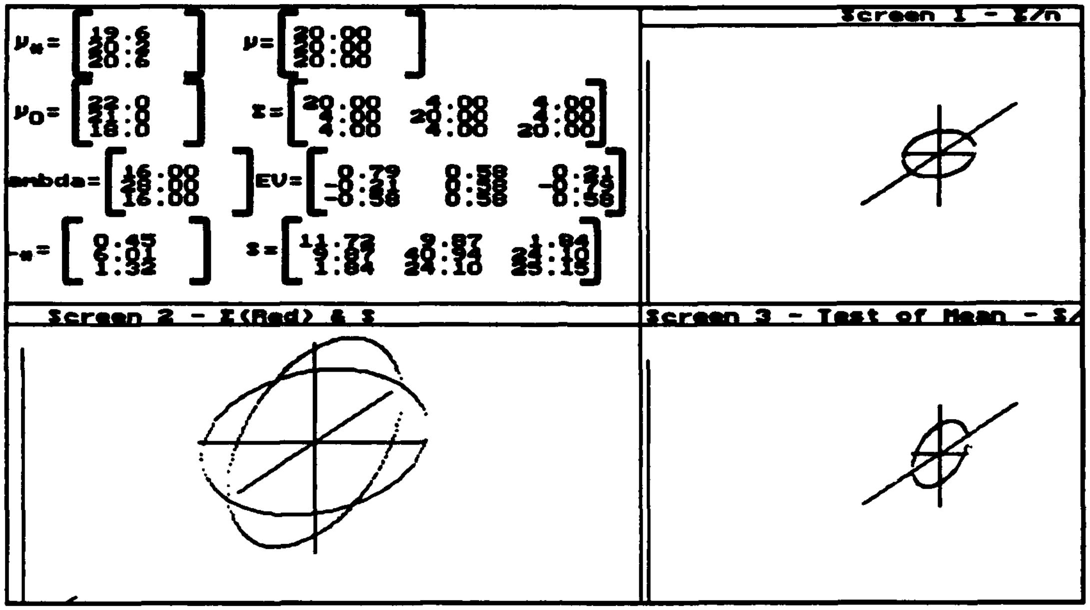


Figure A.23 Output Screen for Case 2 Option 2A

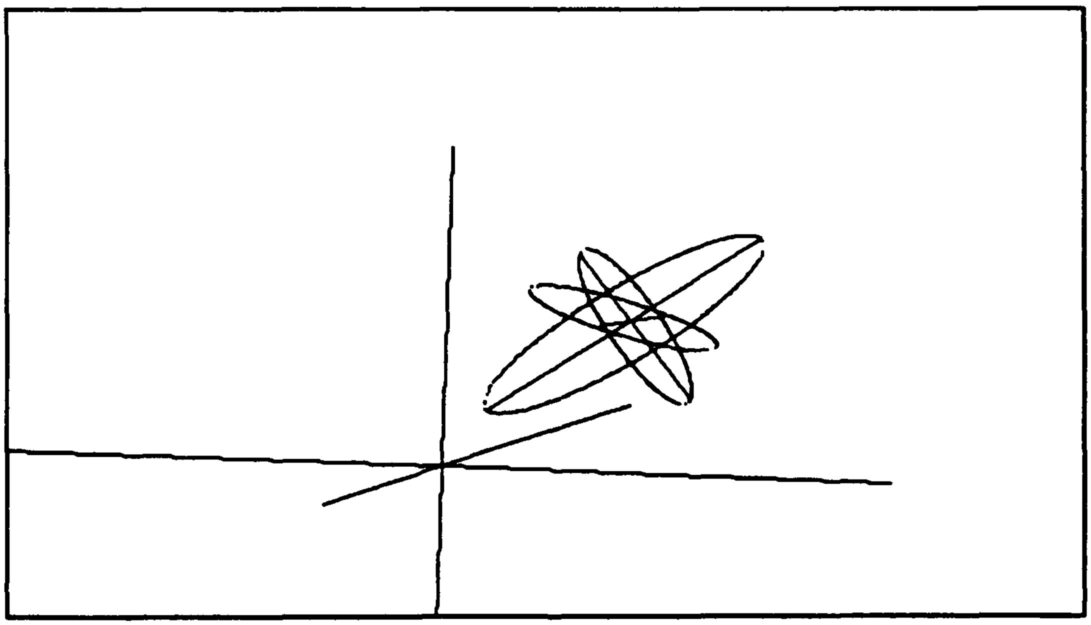


Figure A.24 Output for Case 2 Option 2A Focus on Screen 2

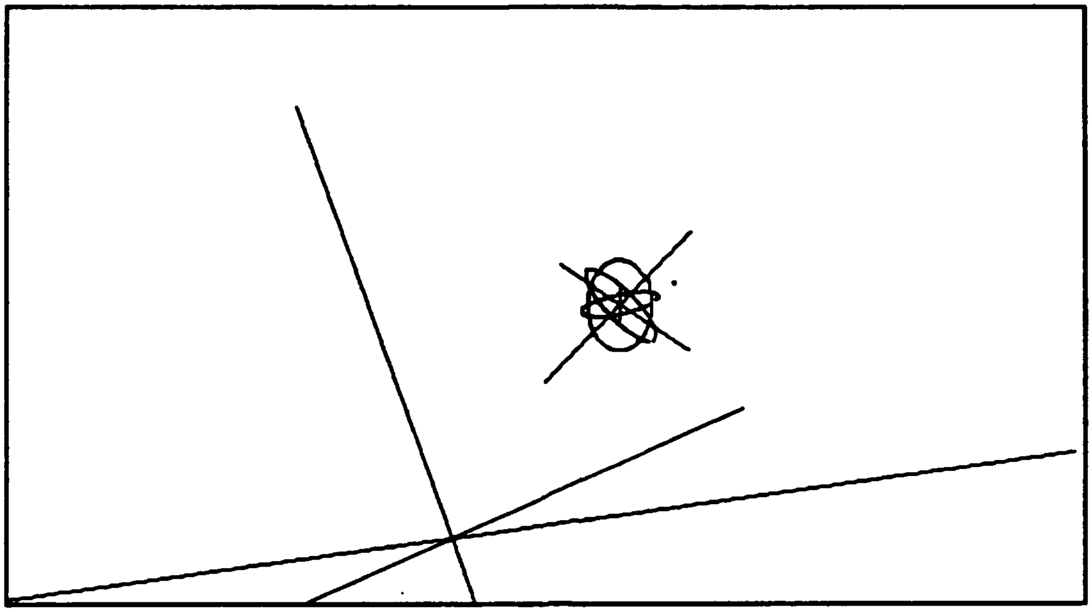


Figure A.25 Output for Case 2 Option 2A Focus on Screen 3

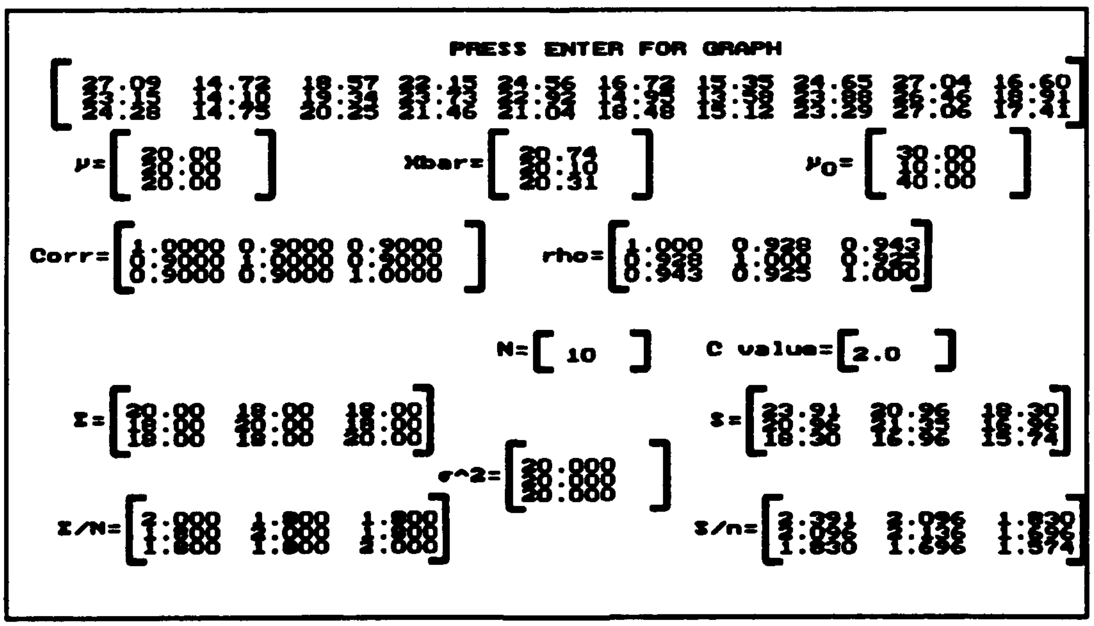


Figure A.26 Input Screen for Case 2 Option 2B

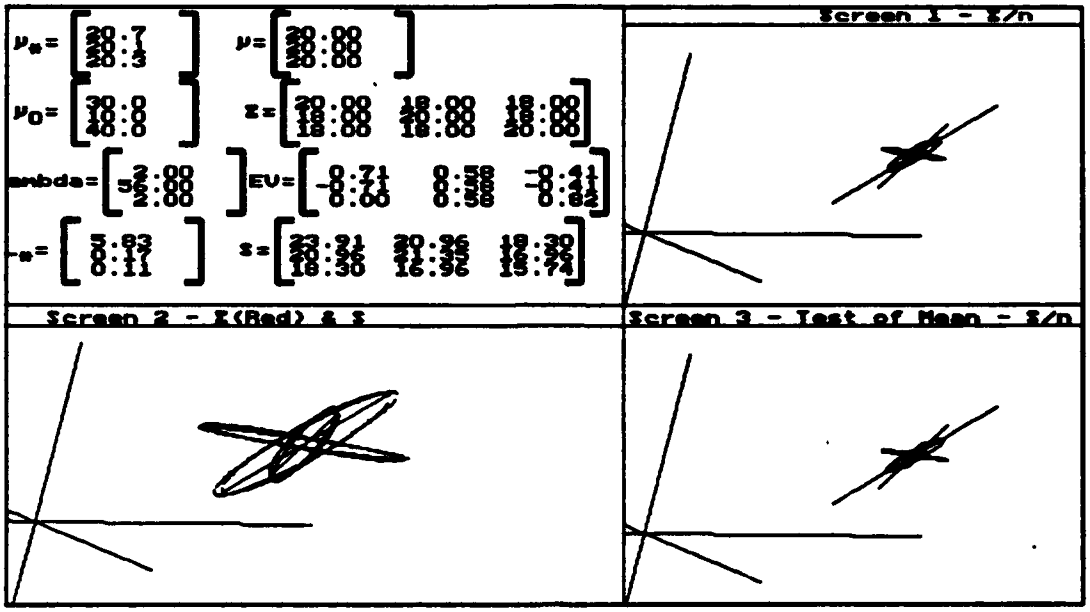


Figure A.27 Output Screen for Case 2 Option 2B

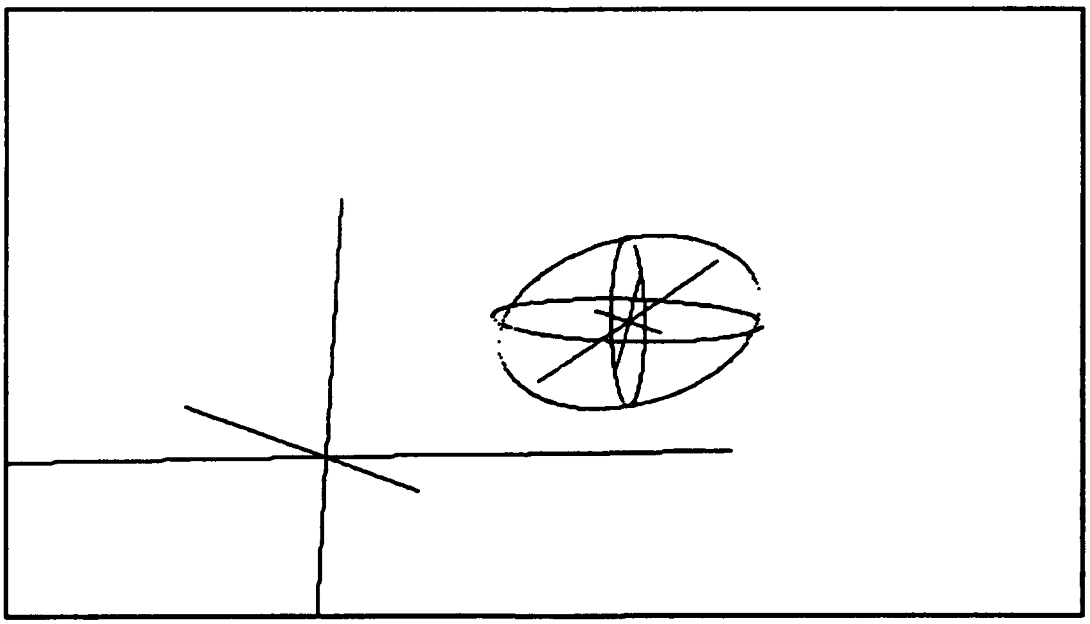


Figure A.28 Output for Case 2 Option 2B Focus on Screen 2

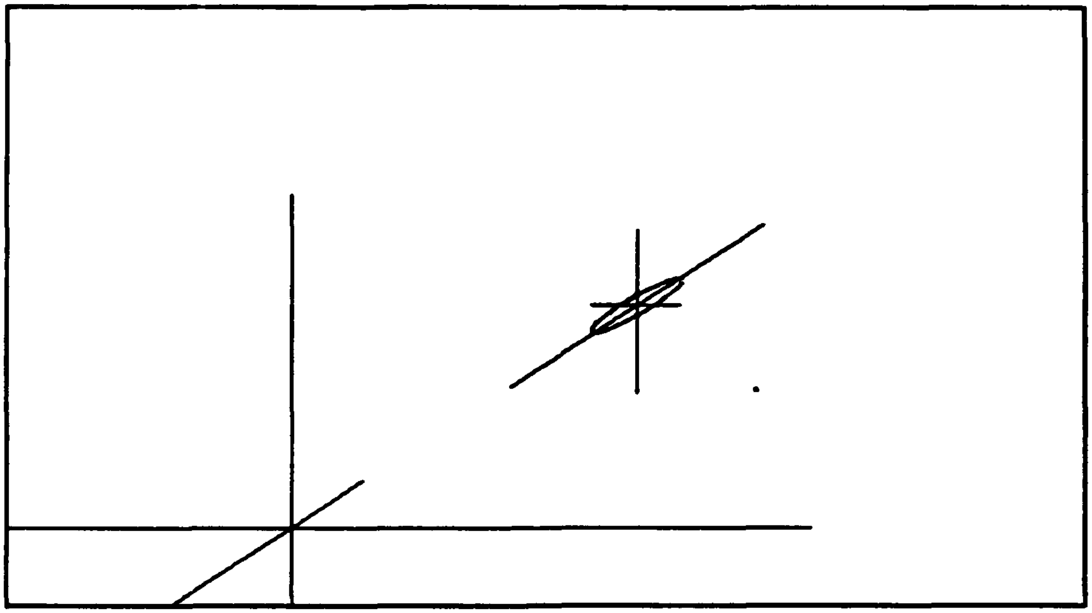


Figure A.29 Output for Case 2 Option 2B Focus on Screen 3

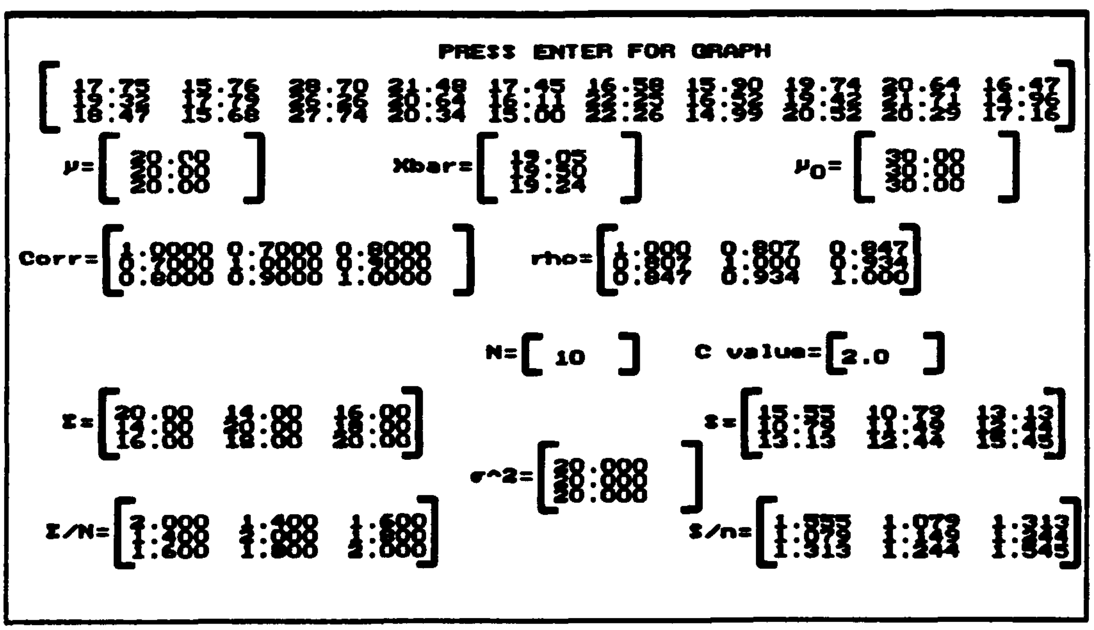


Figure A.30 Input Screen for Case 2 Option 2C

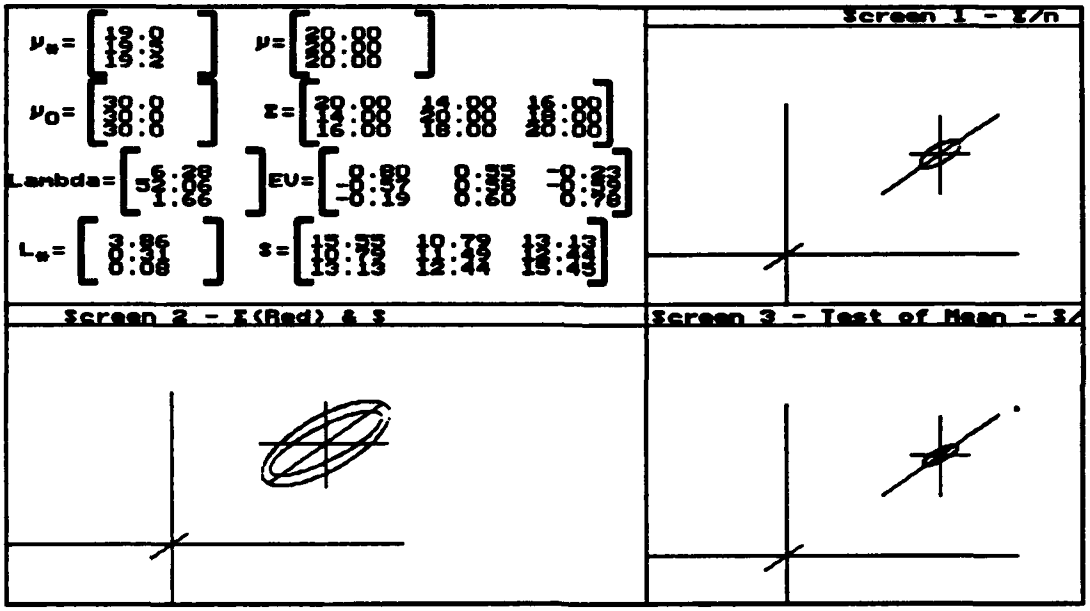


Figure A.31 Output for Case 2 Option 2C

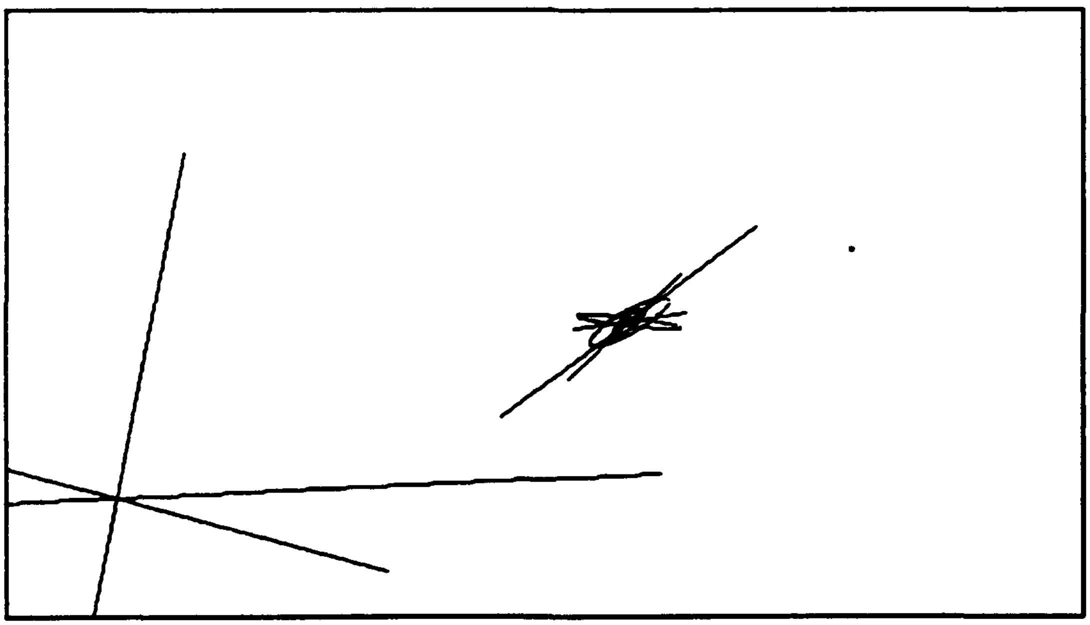


Figure A.32 Output for Case 2 Option 2C Focus on Screen 3

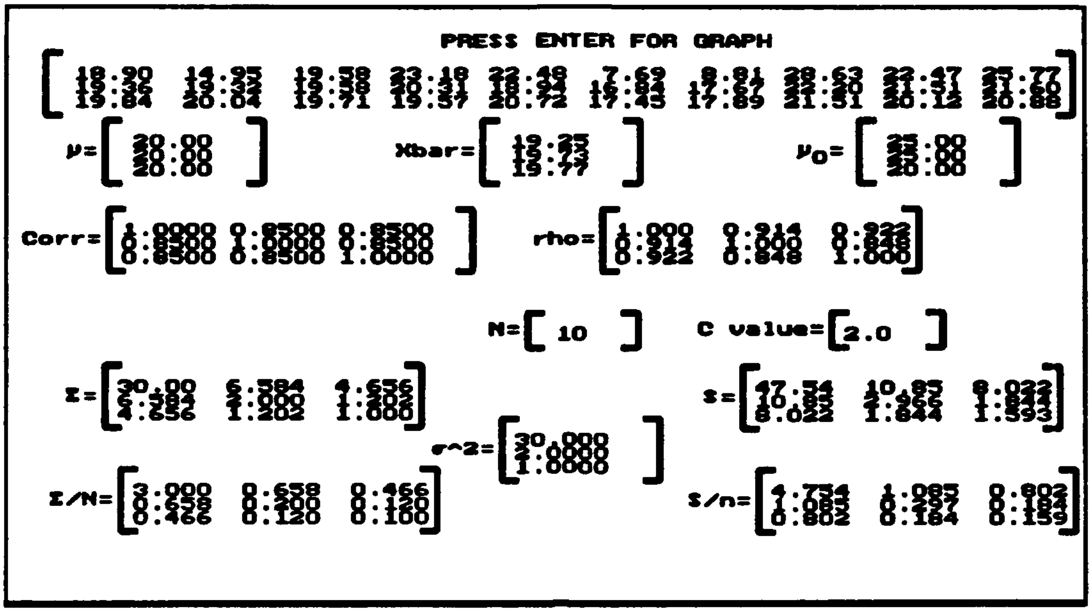


Figure A.33 Input Screen for Case 2 Option 3

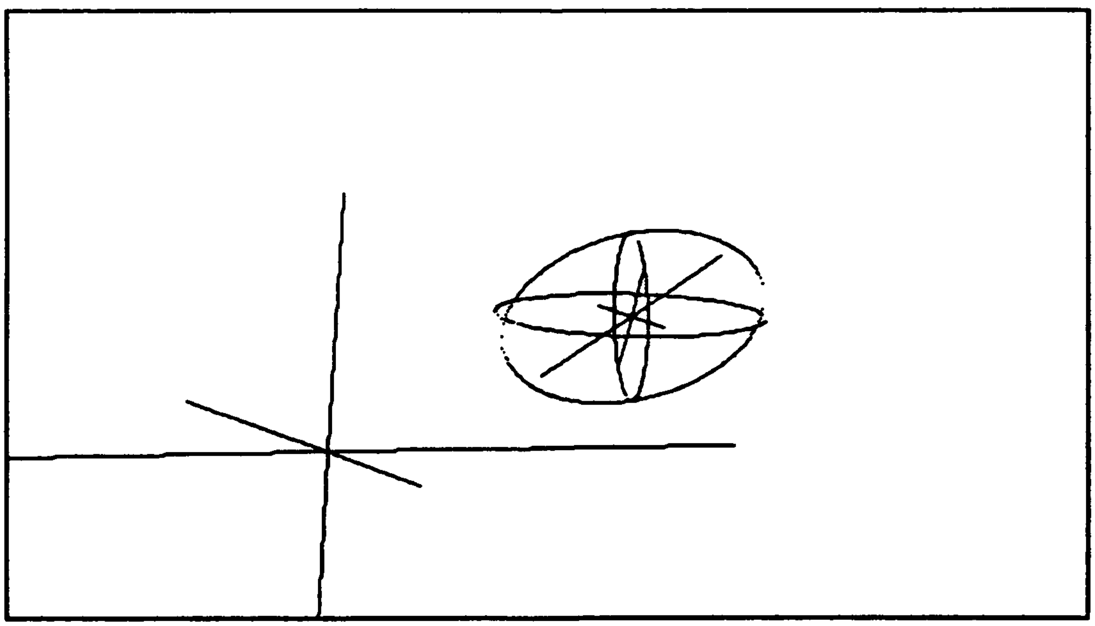


Figure A.34 Output for Case 2 Option 2B Focus on Screen 2

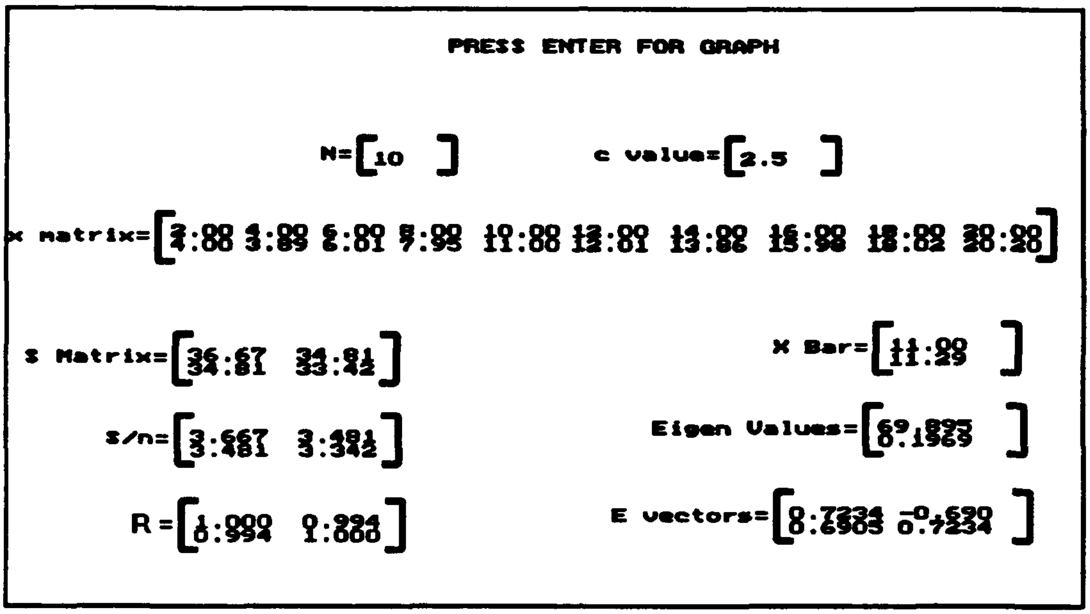


Figure A.35 Input Screen for Case 3 Option 1

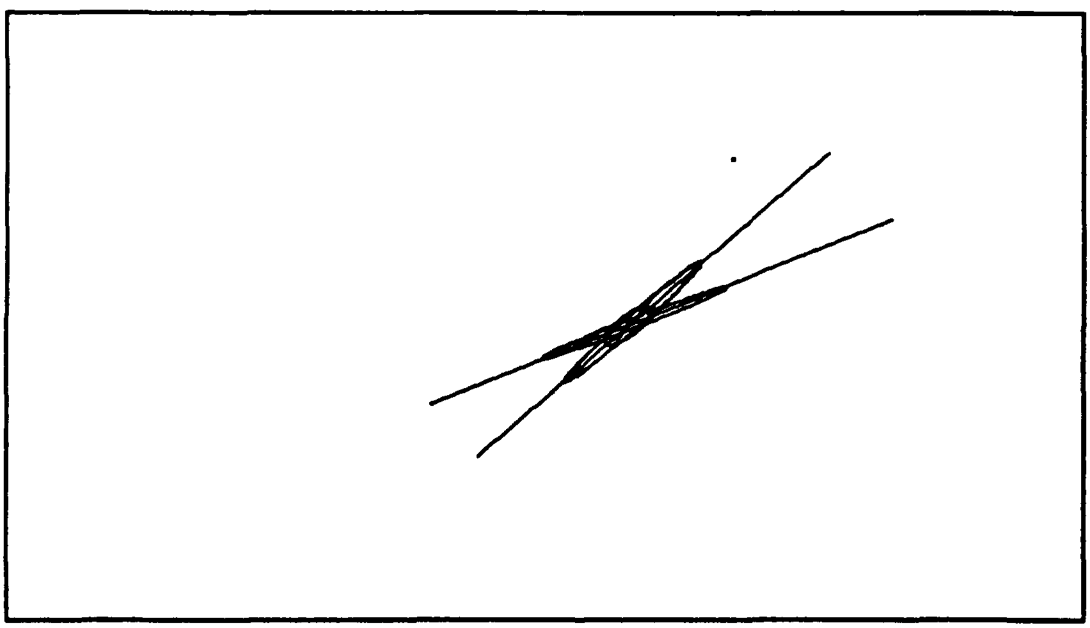


Figure A.36 Output for Case 2 Option 3 Focus on Screen 3

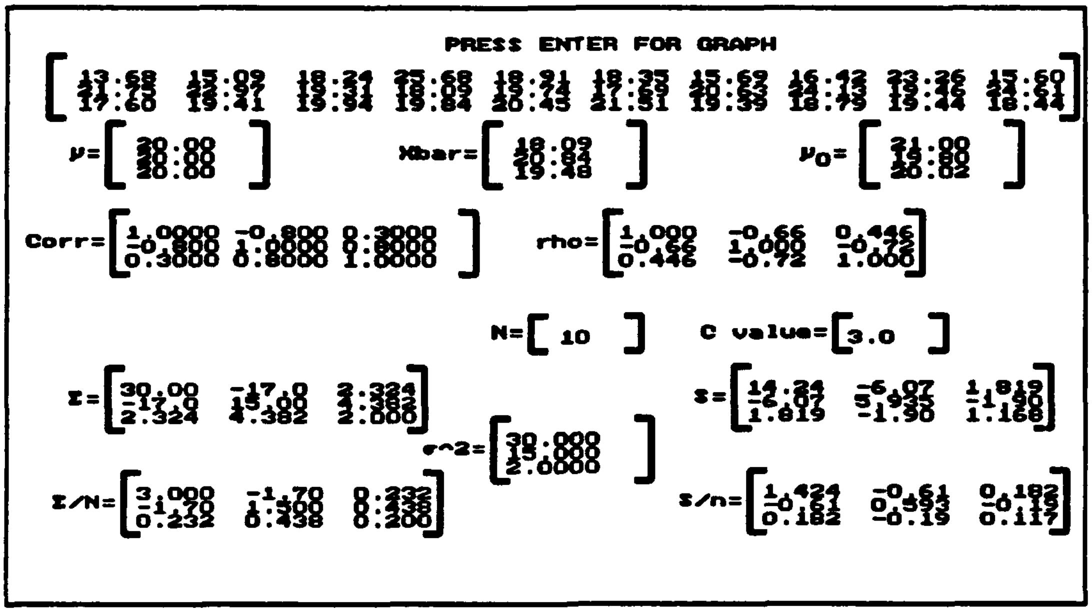


Figure A.37 Input Screen for Case 2 Option 4A

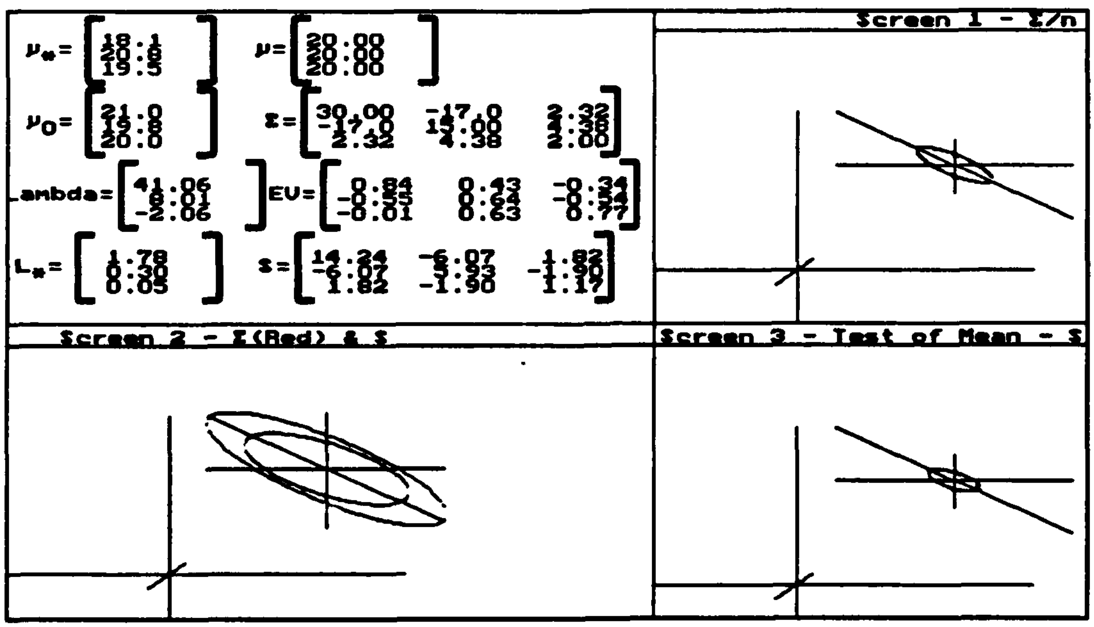


Figure A.38 Output for Case 2 Option 4A

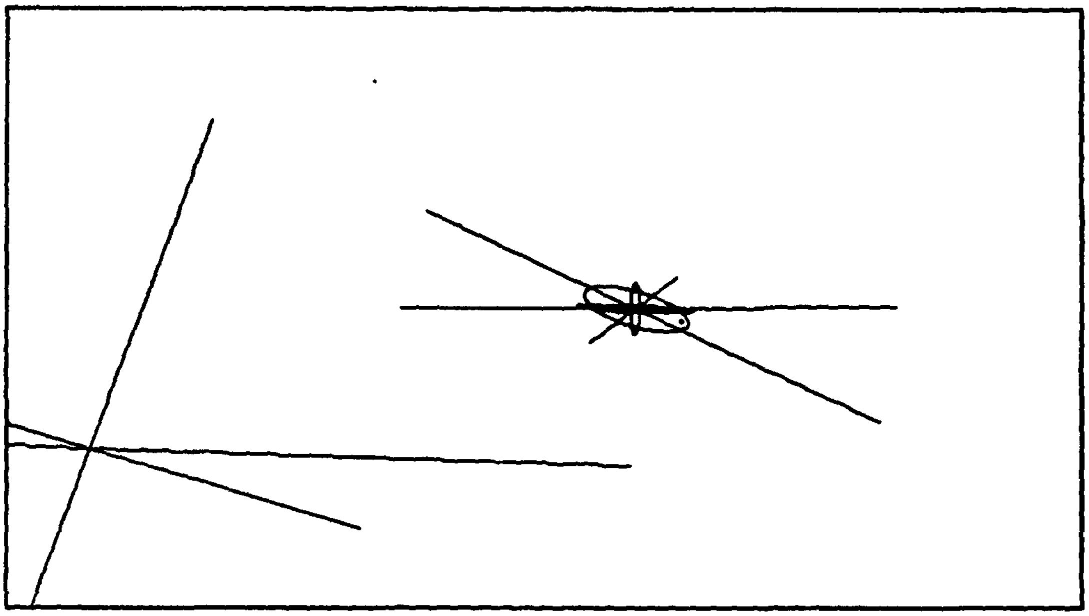


Figure A.39 Output for Case 2 Option 4A Focus on Screen 3

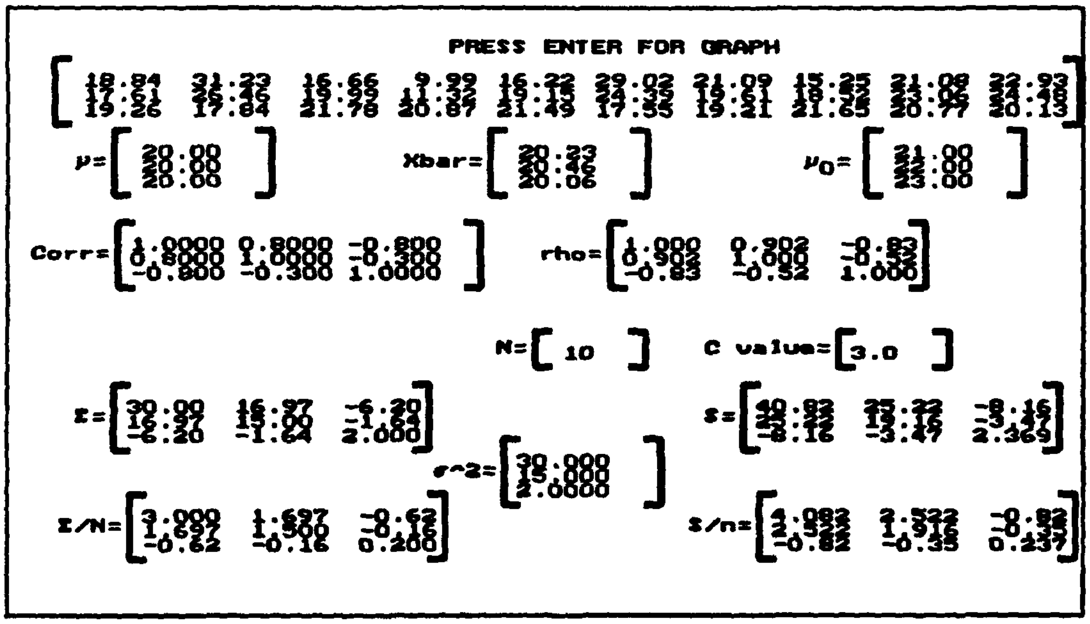


Figure A.40 Input Screen for Case 2 Option 4B

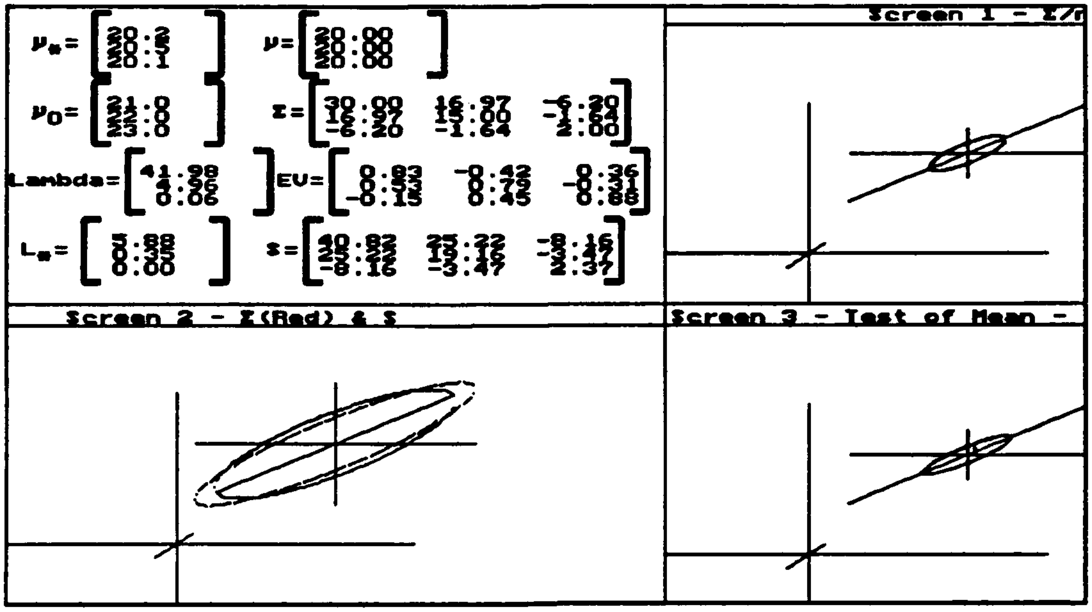


Figure A.41 Output Screen for Case 2 Option 4B

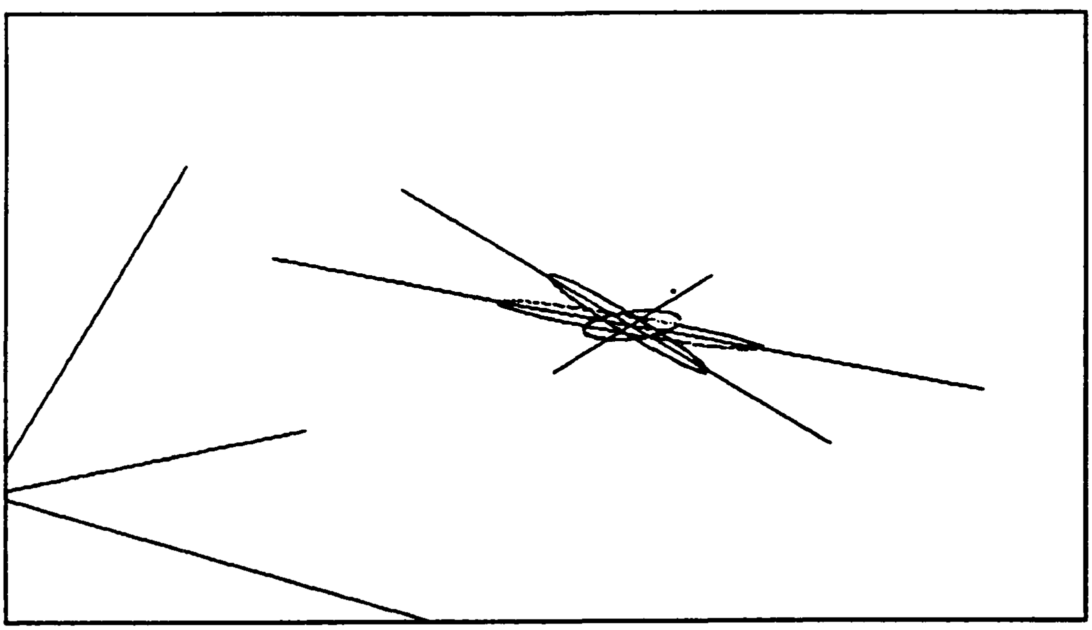


Figure A.42 Output for Case 2 Option 4B Focus on Screen 3

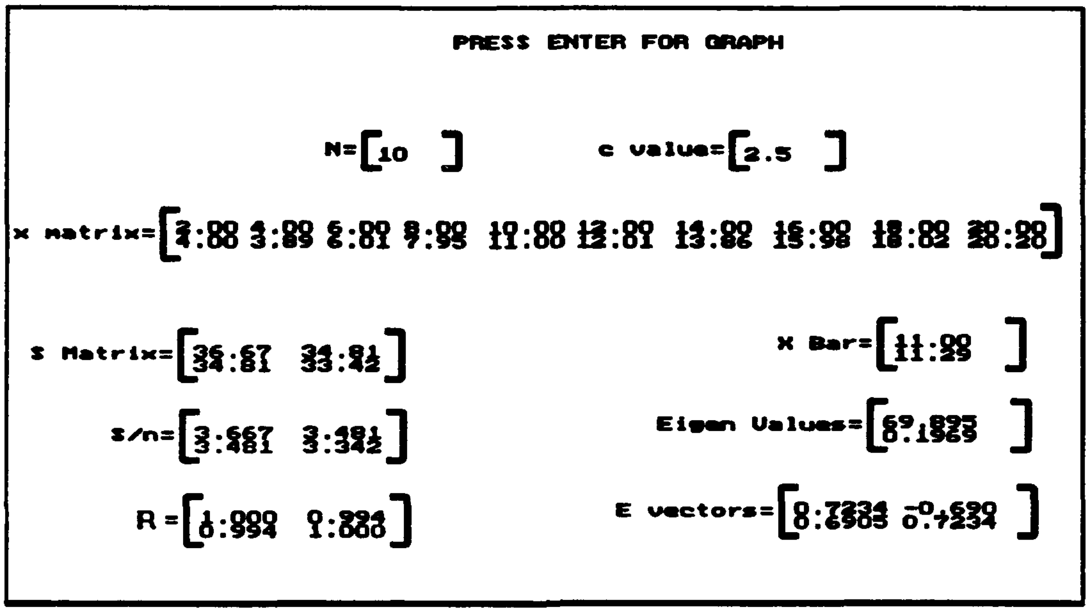


Figure A.43 Input Screen for Case 3 Option 1

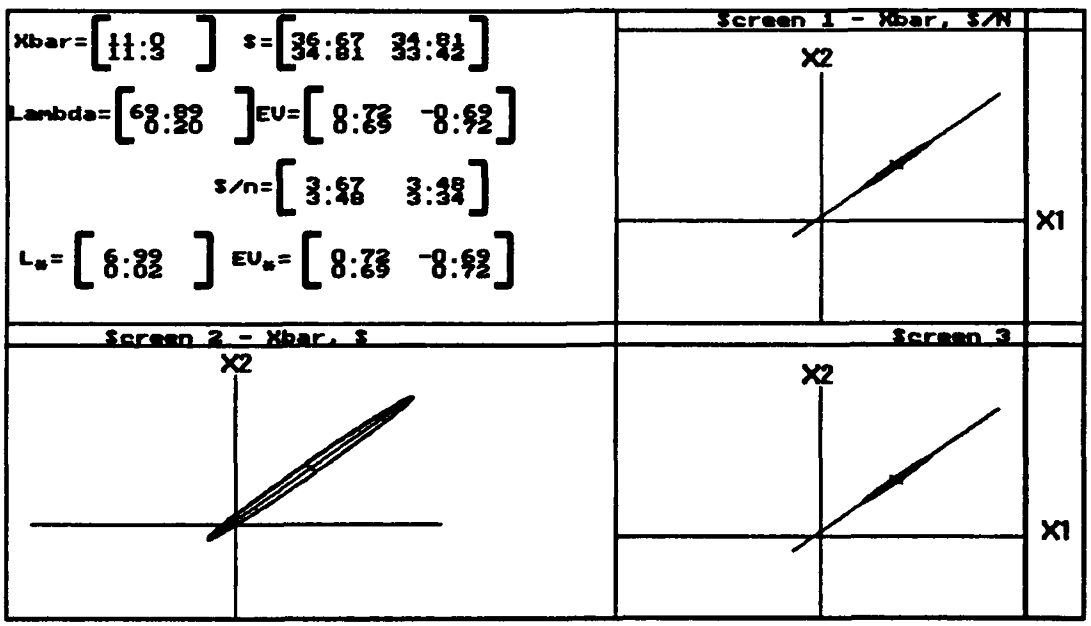


Figure A.44 Output for Case 3 Option 1

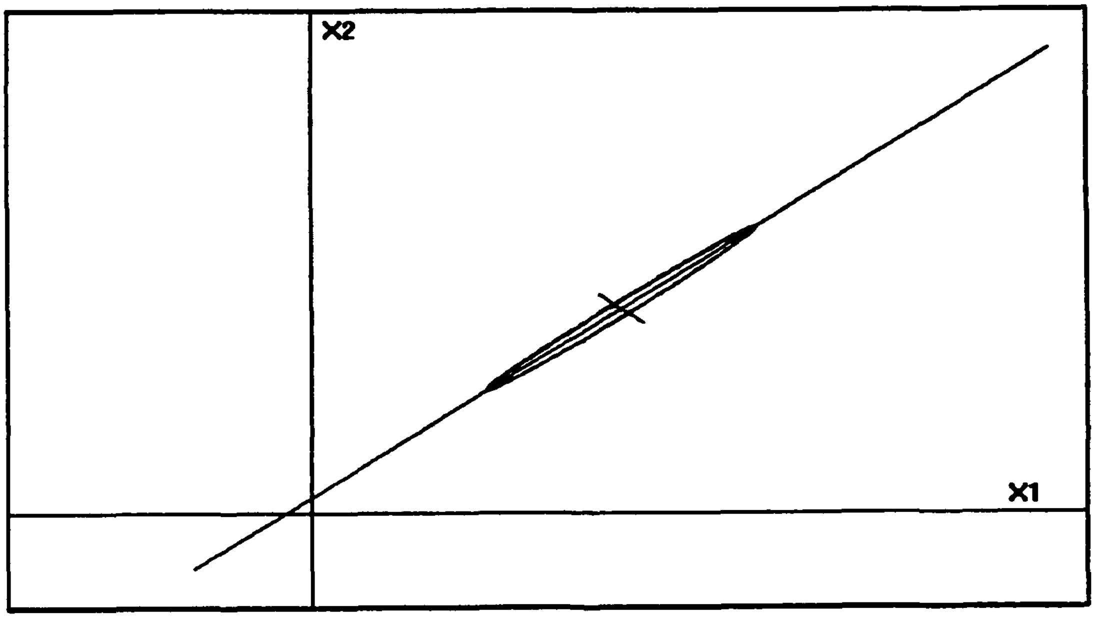


Figure A.45 Output for Case 3 Option 2 Focus on Screen 1

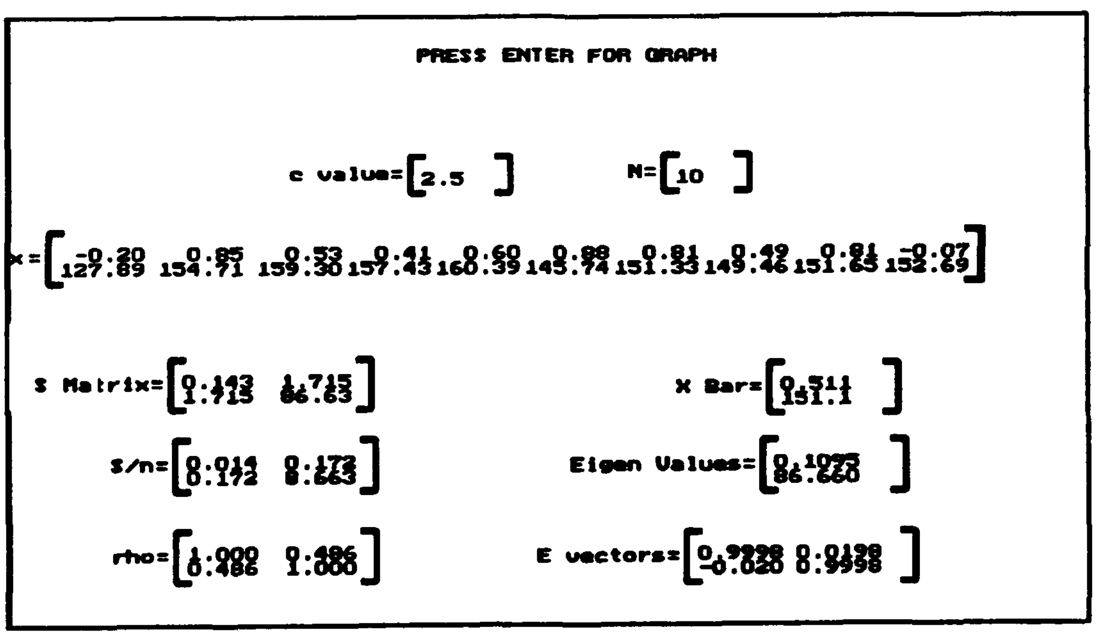


Figure A.46 Input Screen for Case 3 Option 2

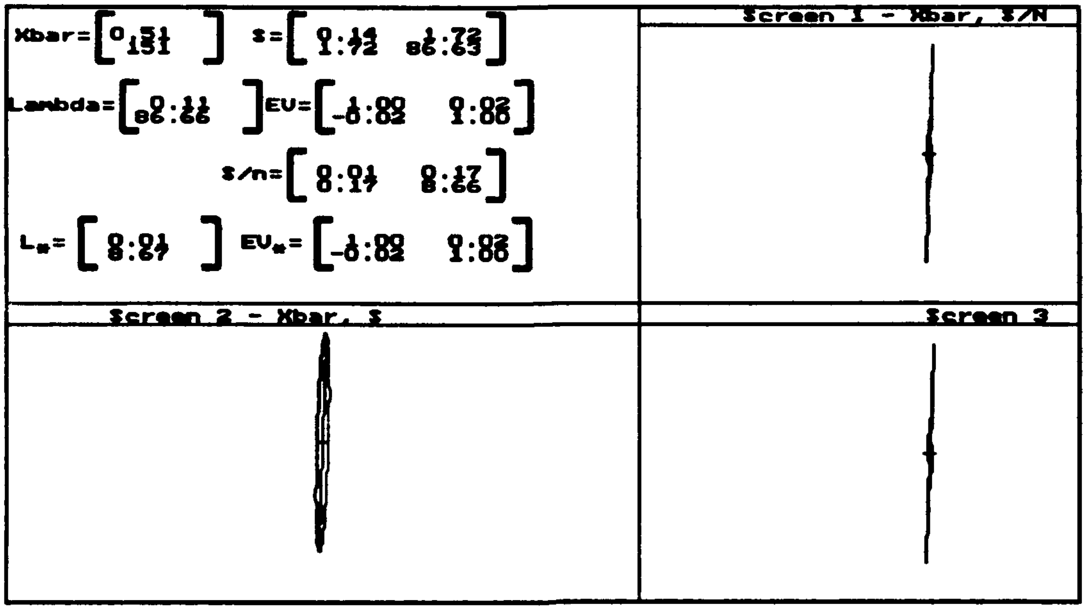


Figure A.47 Output for Case 3 Option 2

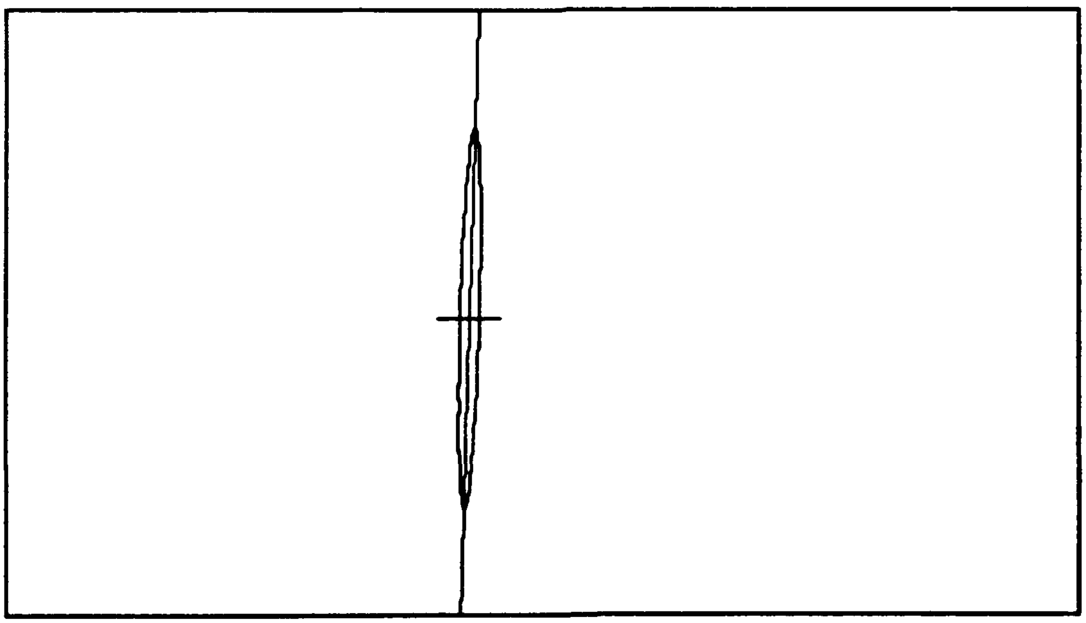


Figure A.48 Output for Case 3 Option 2 Focus on Screen 1

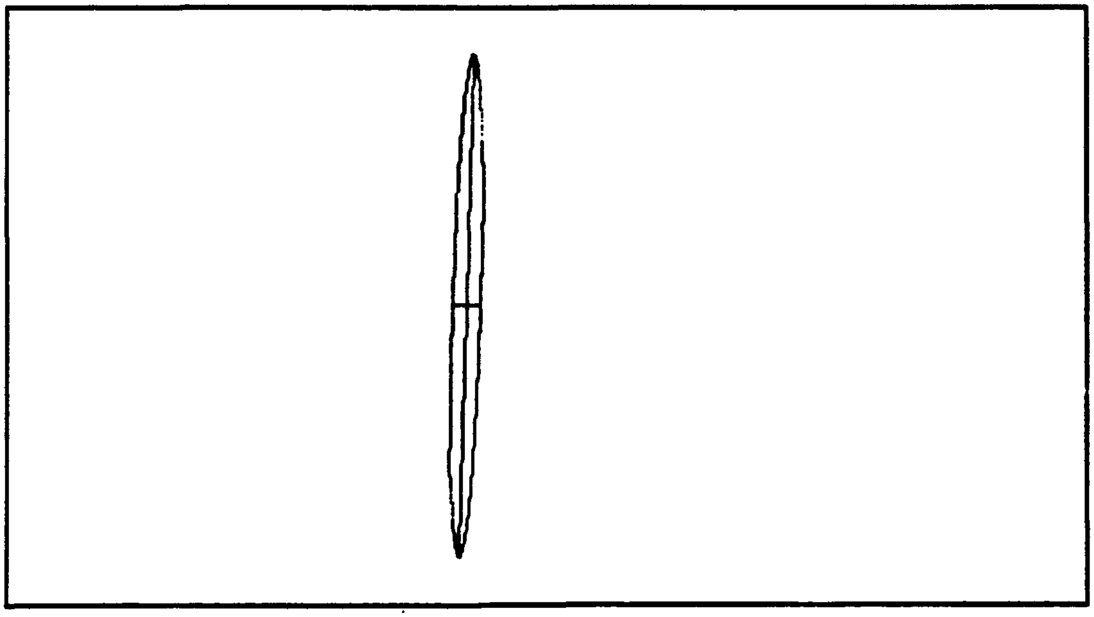


Figure A.49 Output for Case 3 Option 2 Focus on Screen 2

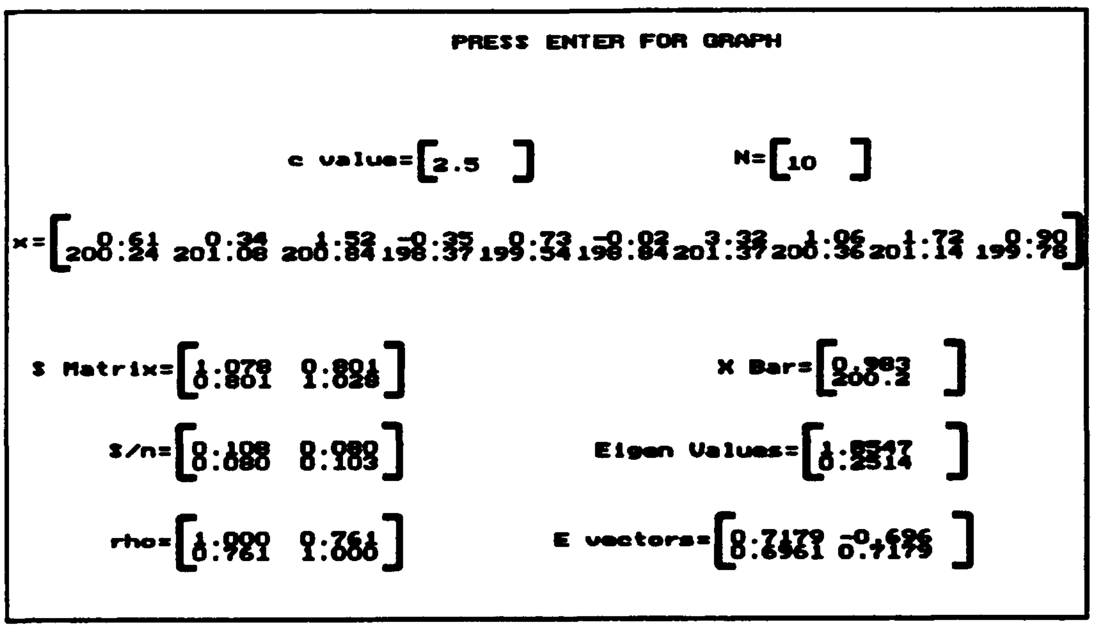


Figure A.50 Input Screen for Case 3 Option 3

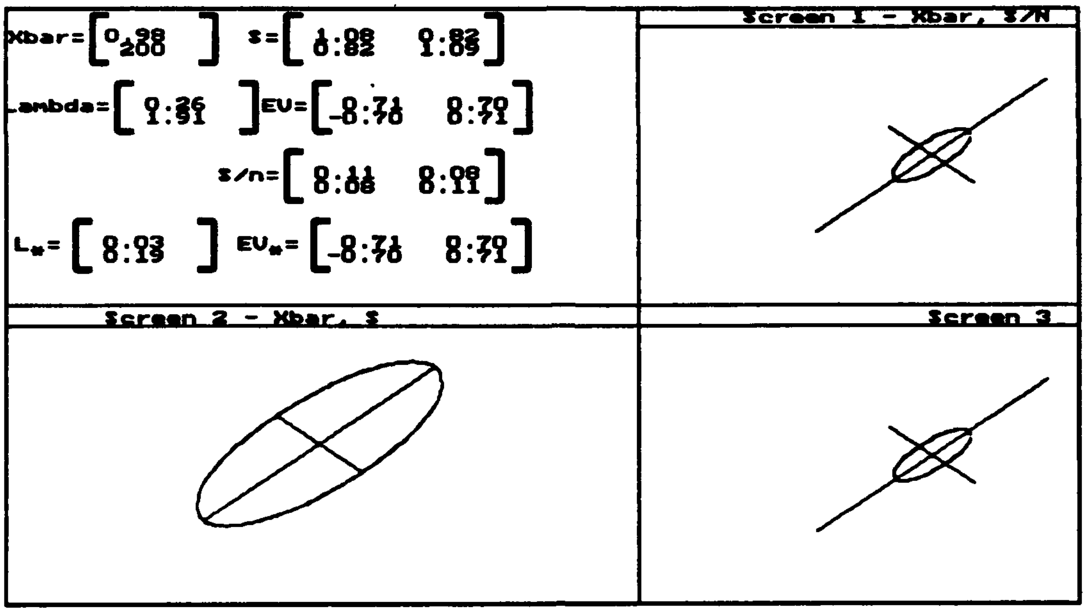


Figure A.51 Output for Case 3 Option Option 3 Enlarged View

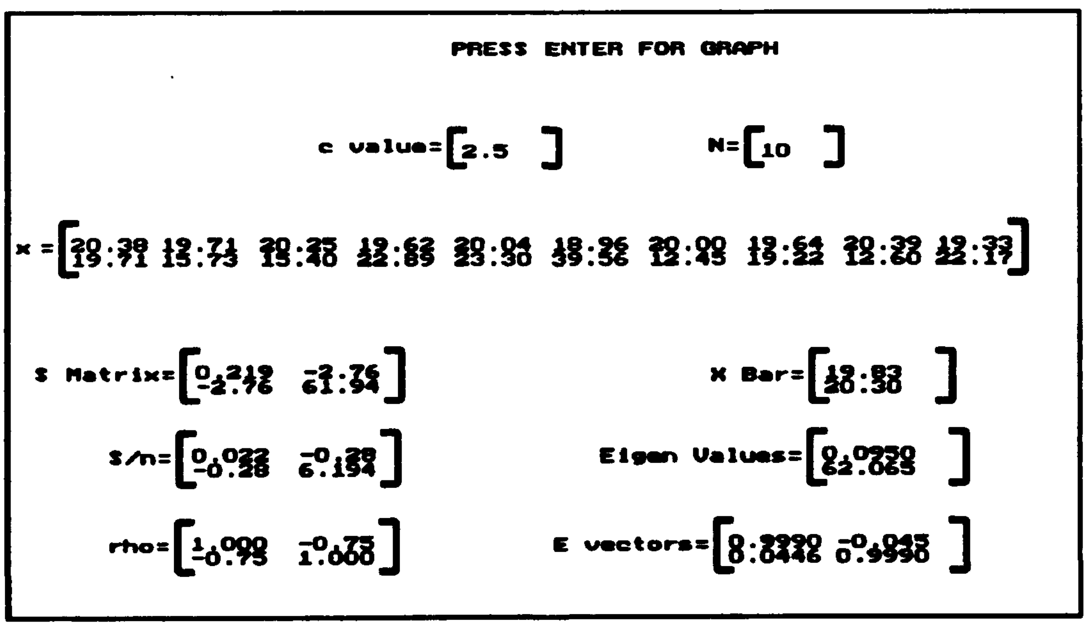


Figure A.52 Input Screen for Case 3 Option 4A

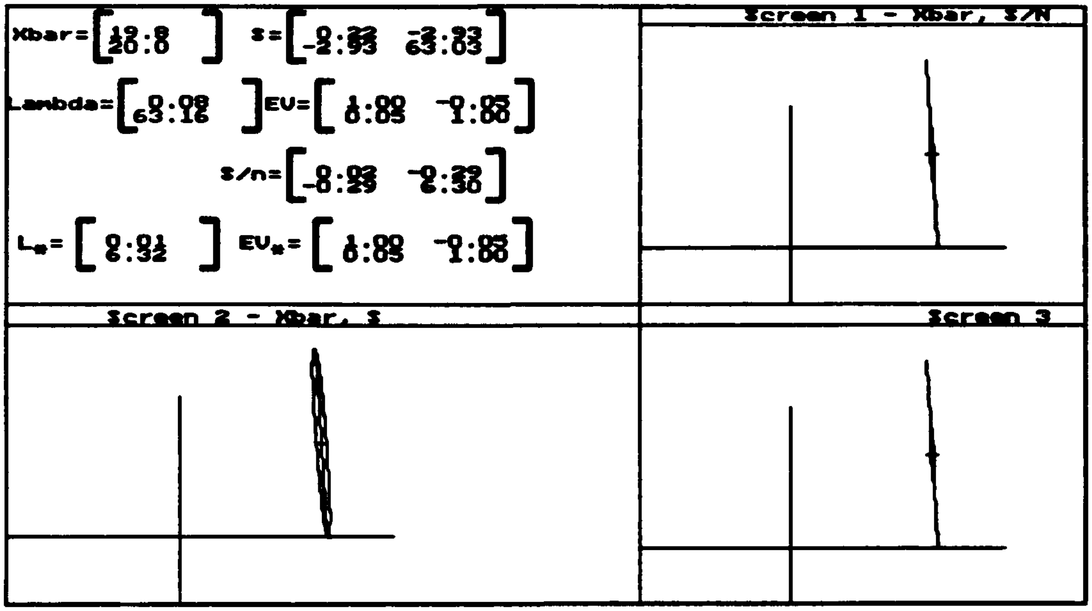


Figure A.53 Output for Case 3 Option 4A

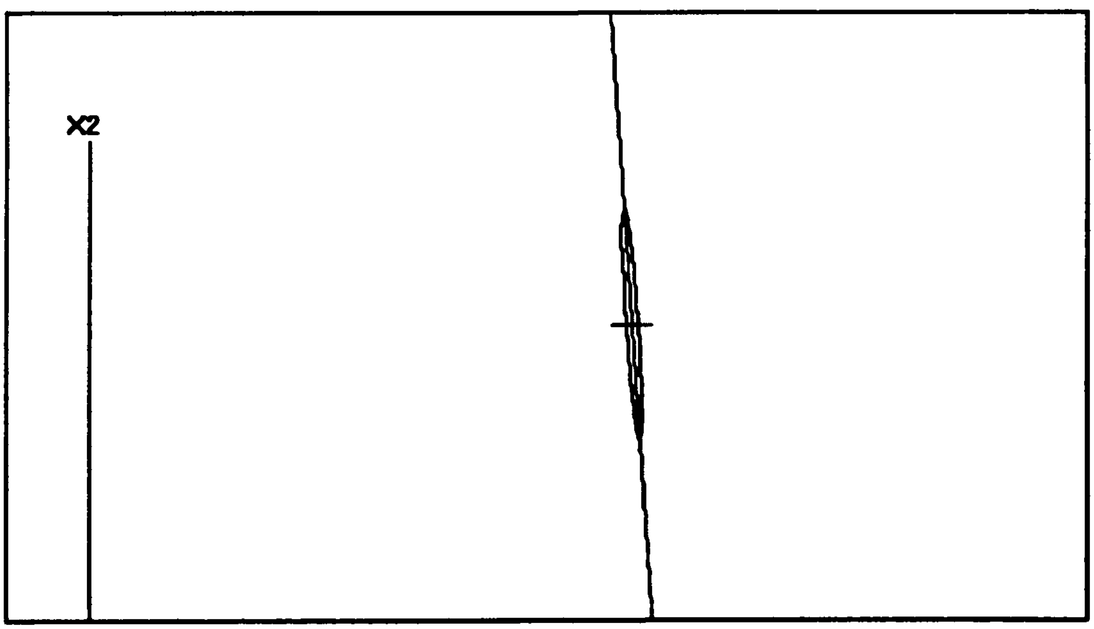


Figure A.54 Output for Case 2 Option 4A Focus on Screen 1

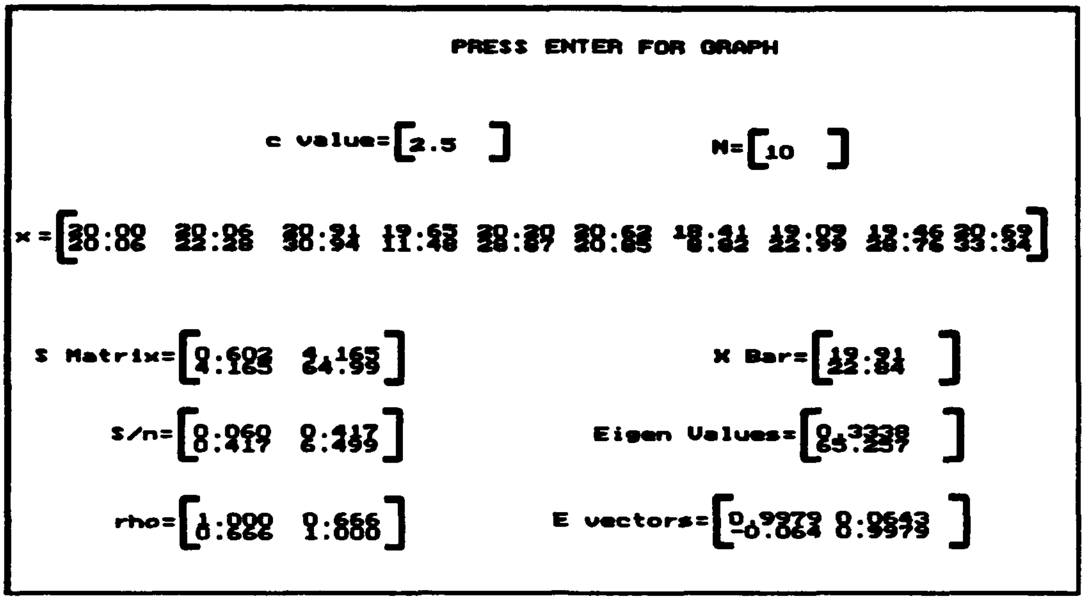


Figure A.55 Input Screen for Case 3 Option 4B

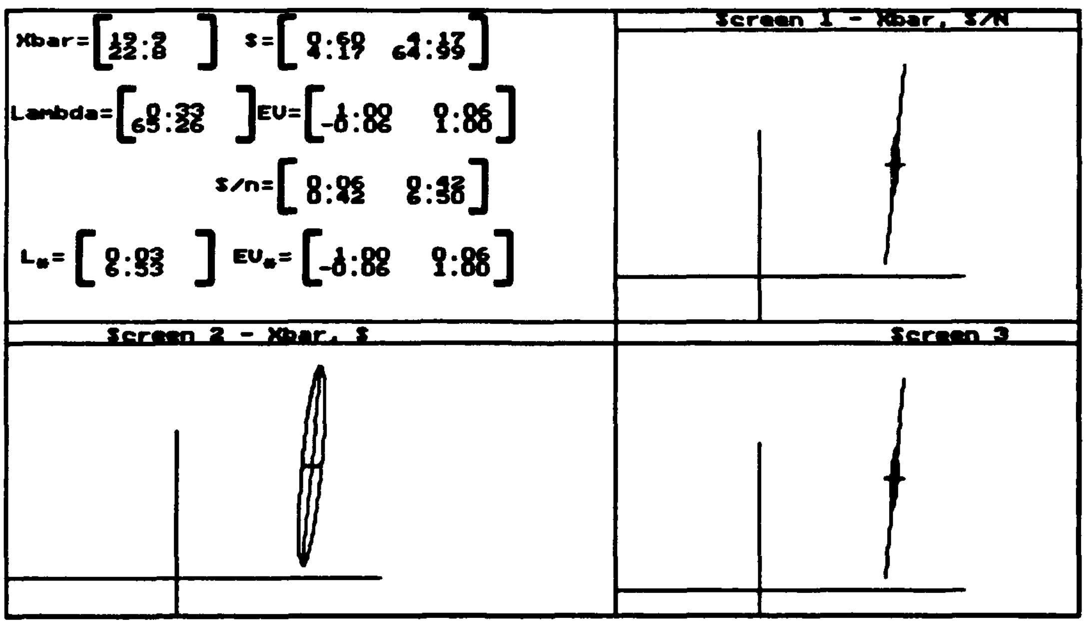


Figure A.56 Output for Case 3 Option 4B

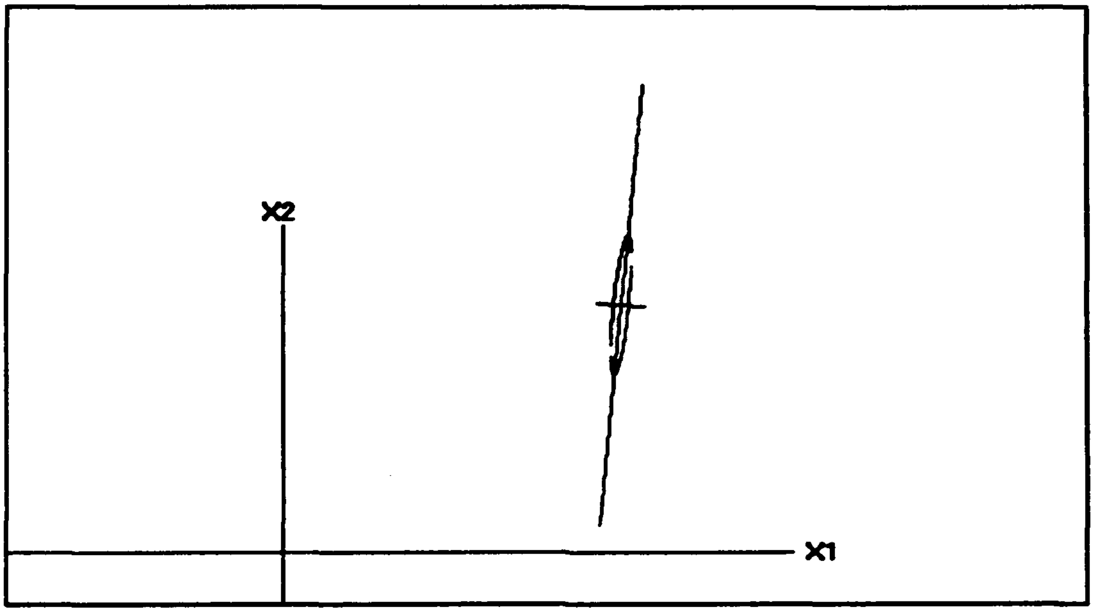


Figure A.57 Output for Case 3 Option 4B Focus on Screen 1

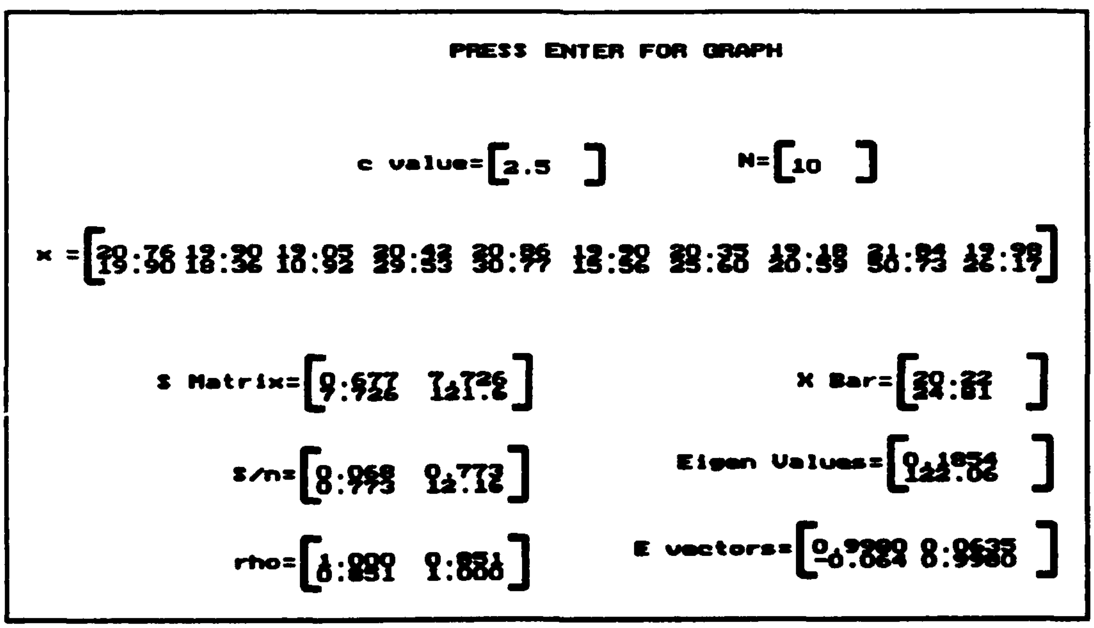


Figure A.58 Input Screen for Case 3 Option 4C

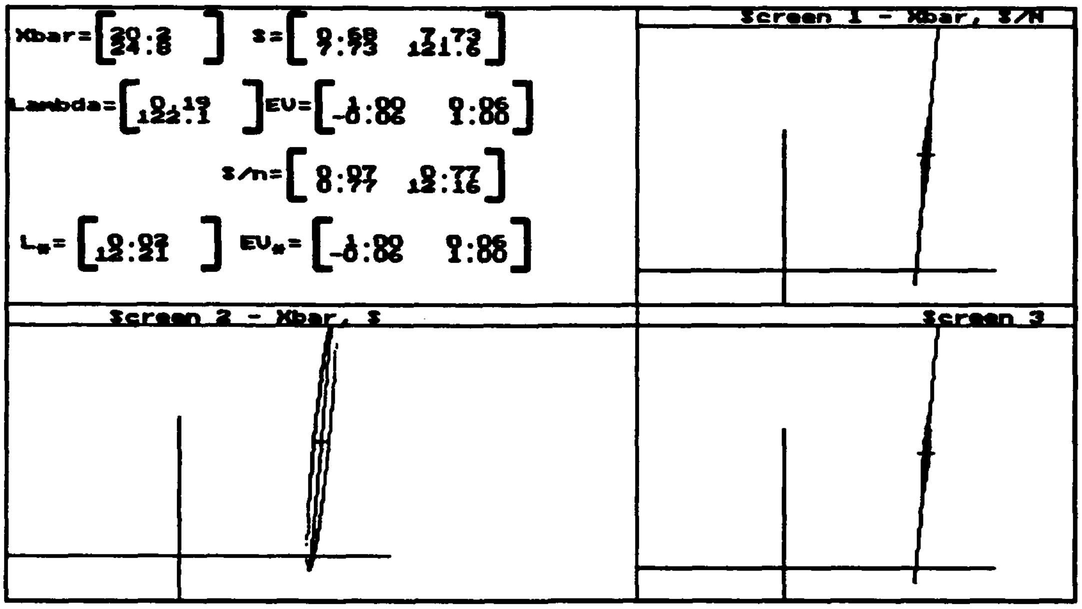


Figure A.59 Output for Case 3 Option 4C

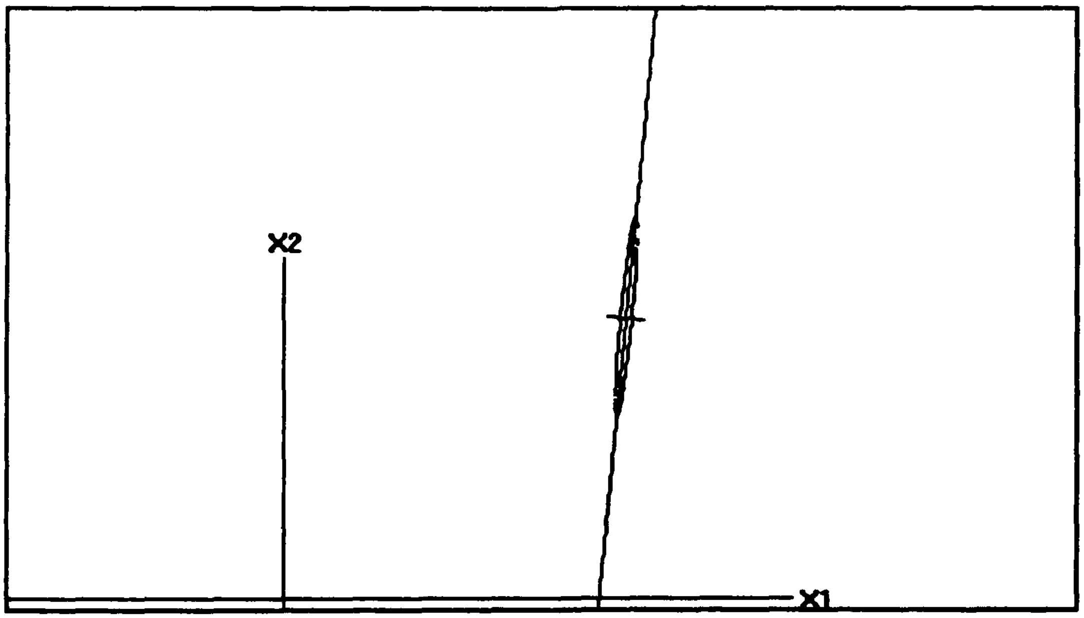


Figure A.60 Output for Case 3 Option 4C Focus on Screen 1

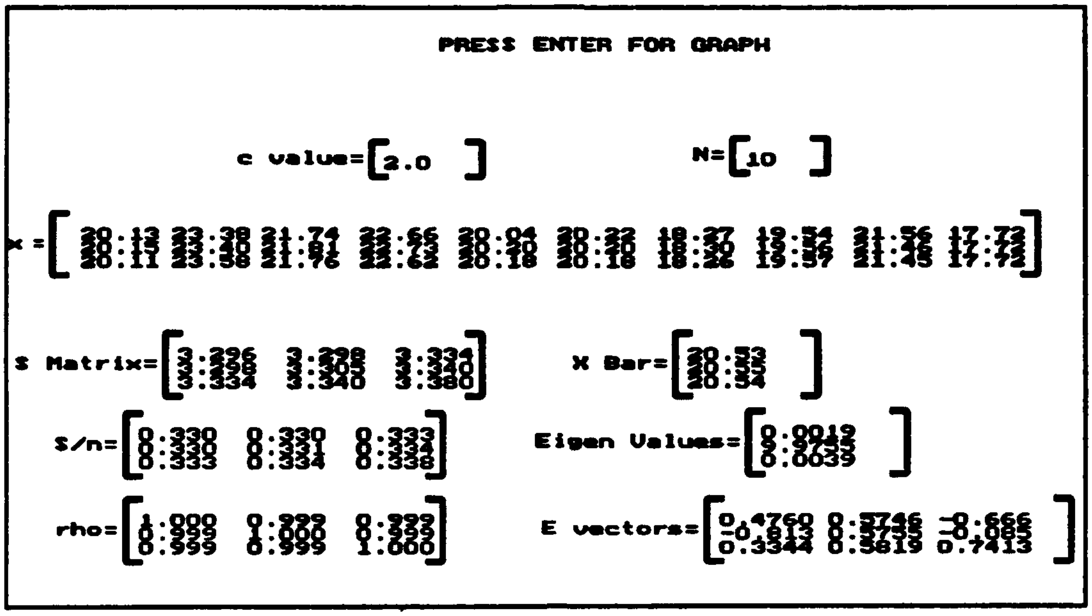


Figure A.61 Input for Case 4 Option 1

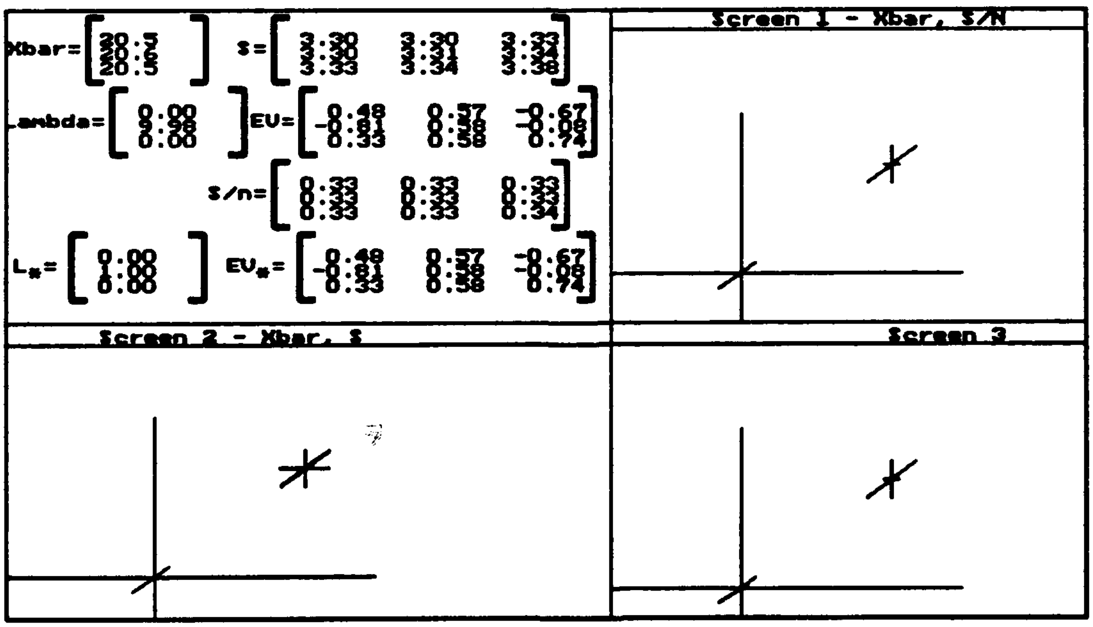


Figure A.62 Output for Case 4 Option 1

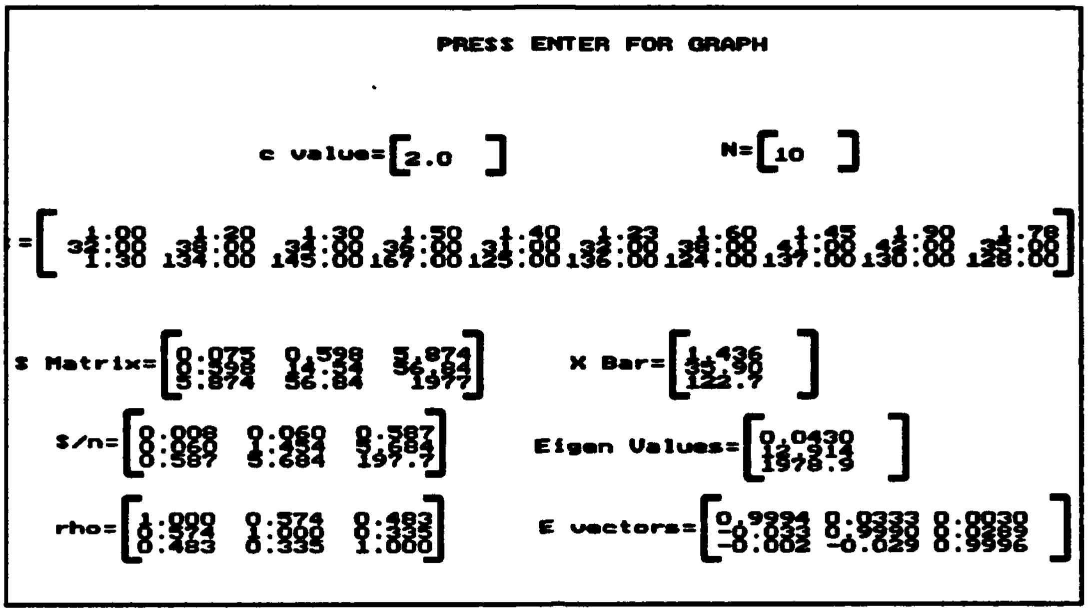


Figure A.63 Input Screen for Case 4 Option 2

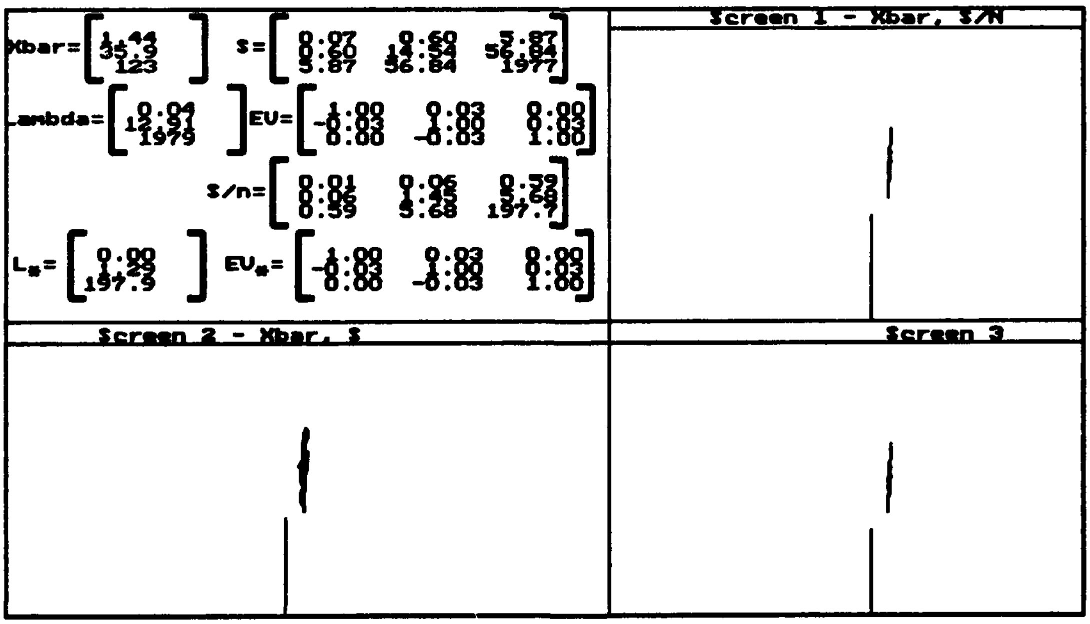


Figure A.64 Output for Case 4 Option 2

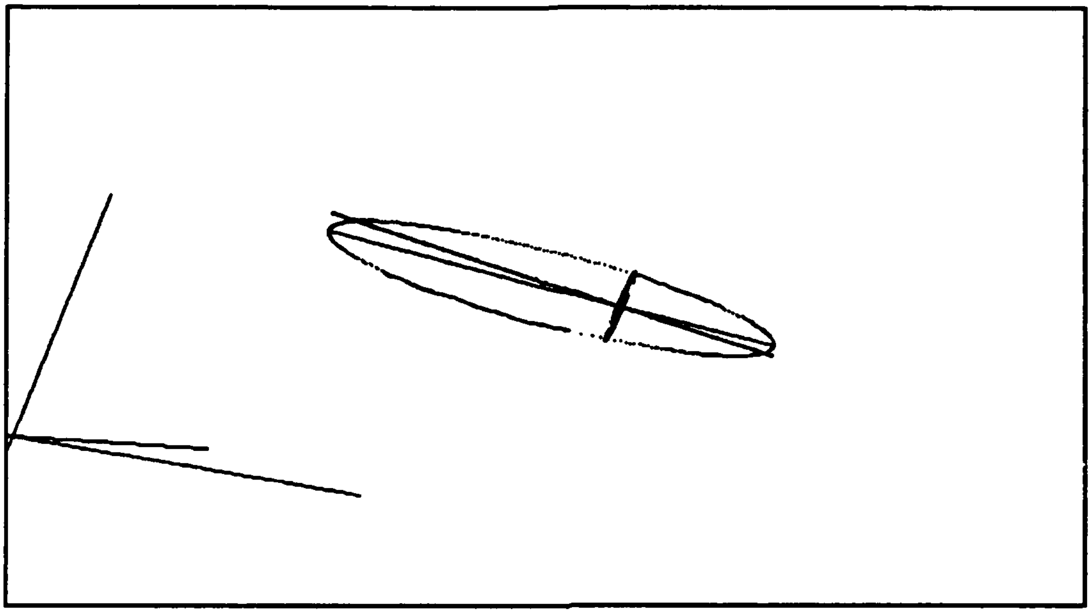


Figure A.65 Output for Case 4 Option 2 Focus on Screen 2

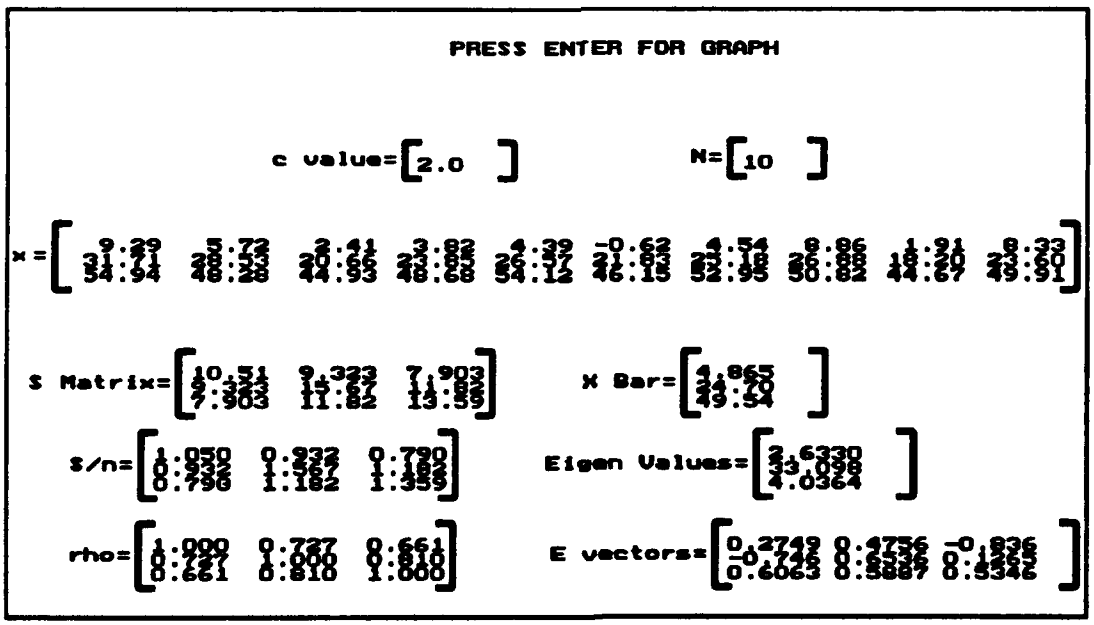


Figure A.66 Input Screen for Case 4 Option 3

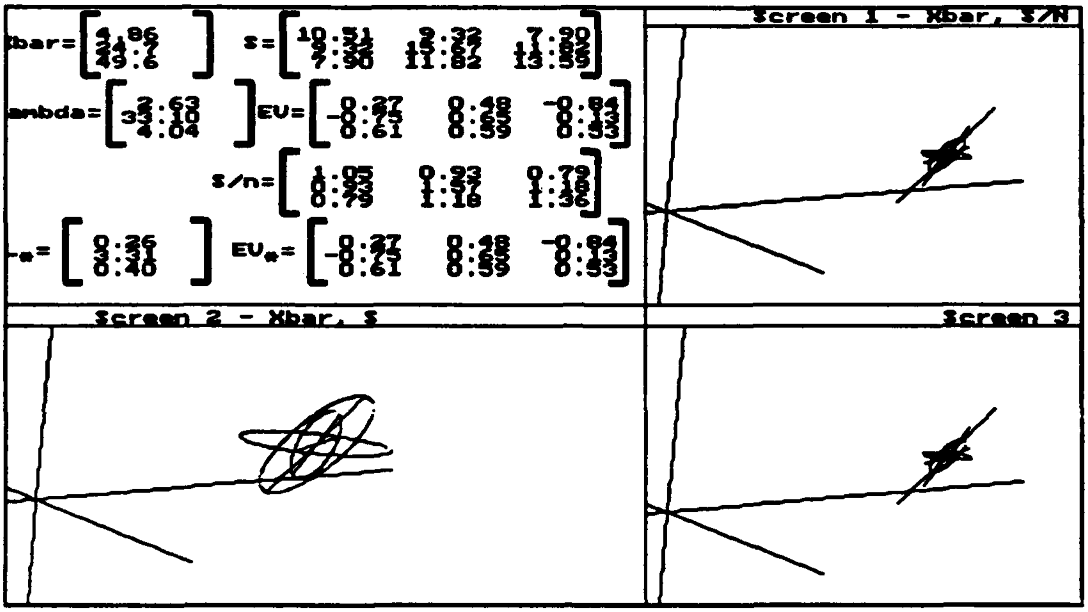


Figure A.67 Output for Case 4 Option 3

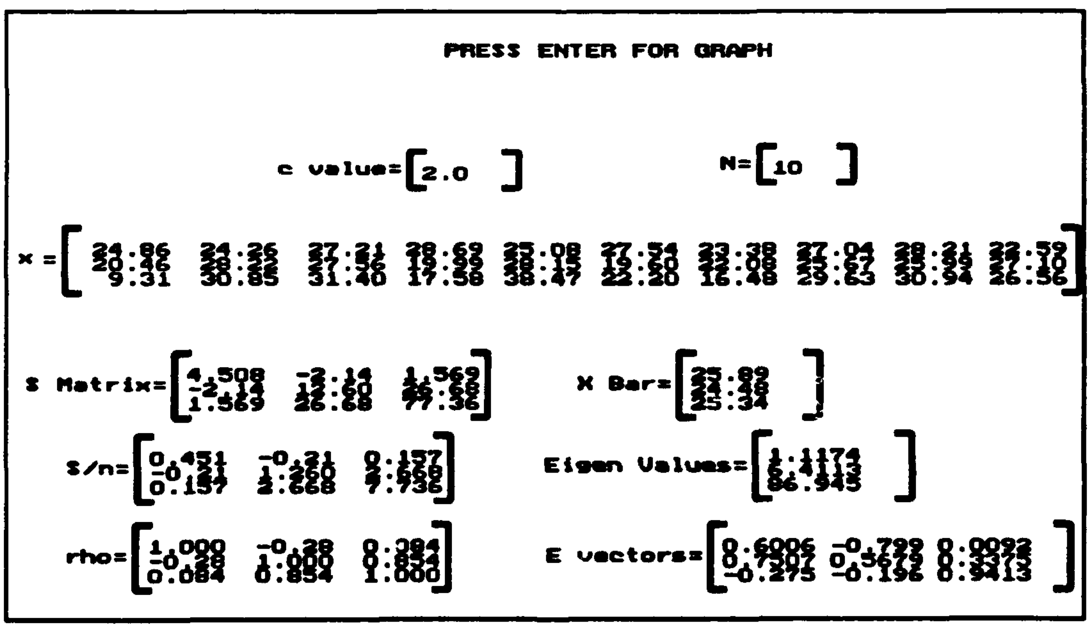


Figure A.68 Input for Case 4 Option 4

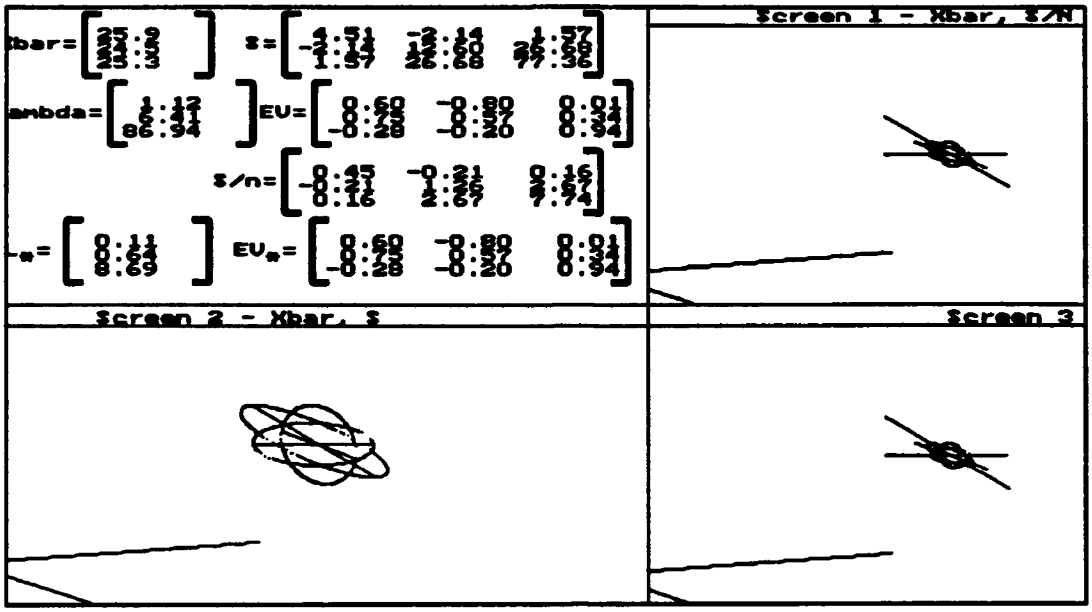


Figure A.69 Output for Case 4 Option 4

Appendix B

The main program contains the following procedures:

1. InitObject = prepares points, endpoints and lines so they are ready to be processed and displayed in AcroMolé.

2. GenerateAxis = generates the coordinate system and the major and, if in 2-D case, the minor axis of the ellipses is displayed.

3. Ellipses = generates the ellipses out of points.

4. DrawOutlines = generates the boarders of the main screen and displays the information needed according to the case being used.

```
Program Ellpise;
{$F+}
(* This program is an example of some of the things that you can do with the  *)
(* AcroMole subroutine library published by Acrobits, USA, (801)966-2580 or   *)
(* toll-free in the US & Canada (800)ACROBITS (i.e. (800)227-6248).     *)

uses Overlay,Mathstuf,mathmat,rgraphma,graph,Get_data,crt;

{$L MOLE9.OBJ}  (* Include object code for AcroMole subroutines. *)
{$I ACROMOLE.PAS} (* Include Pascal definitions for AcroMole subroutines. *)
{$I UTILITY.PAS}  (* Include Pascal utility routines. *)
{$O Mathstuf}
{$O mathmat}
{$O rgraphma}
{$O Get_data}

Var
  CalculateScaleFactorsVar:CalculateScaleFactorsRecord;
  DrawLineVar:DrawLineRecord;
  DrawRectangleVar:DrawRectangleRecord;
  SetRight3DCameraVar:Set3DCameraRecord;
  SetLeft3DCameraVar:Set3DCameraRecord;
  SetDrawingBufferVar:SetDrawingBufferRecord;
  SetLeftWindowVar:SetWindowRecord;
  SetUpperWindowVar:SetWindowRecord;
  SetLowerWindowVar:SetWindowRecord;
  SetFullWindowVar:SetWindowRecord;
  TempWorldPoint:WorldPointPointer;
  Endpoint1,Endpoint2:WorldEndpointpointer;
  TempWorldLine:WorldLinePointer;
  I,J,K:Word;
  Temp:Integer;
  WindowWidth:Integer;
  WindowHeight:Integer;
```

```
  WindowHorizontal:Integer;
  WindowVertical:Integer;
  NumberOfBuffers:Word;
  Angle:Real;
  TempReal:Real;
  SinAngle,CosAngle,SinPositiveAngle:Real;
  DeltaDistance:Real;
   xdata,S,Sigma,cvalue,xbar,Rmatrix,eigenvalues,eigenvectors,SigmaByN,
   evalstar,evecstar,Mu,SbyN,txbar:mathmat.matx;
  xhigh,xlow:RealArrayMPbyMP;
  answer:real;
  look,flagf1,flagf2,flagf3:boolean;
  GD,Gm,Errcode:integer;
  choice,P:string;
  ordchoice,code,ordP:integer;
  xx,change,xmax,ymax,zmax:real;
  x1bar,x2bar,x3bar:string;
  L1,L2,L3:string;
  sig2x1,sig2x2,sig2x3:string;
  kolor:byte;
Label 888;
Const
  WorldPointList:WorldPointPointer=Nil;
  WorldEndpointList:WorldEndpointPointer=Nil;
  WorldLineList:WorldLinePointer=Nil;
  WorldPointListS1:WorldPointPointer=Nil;
  WorldPointListS2:WorldPointPointer=Nil;
  WorldPointListS3:WorldPointPointer=Nil;
  Distance:Real=0.7;
  Speed:Real=0.00003;
  Matrix:Array[0..2,0..2] Of Real=   (* Rotation matrix. *)
    ((32760.0,0.0,0.0),(0.0,32760.0,0.0),(0.0,0.0,32760.0));

(*-------------------------------------------------------------------------*)
(* Initializes an object that has up to 255 endpoints, and      *)
(* 255 lines.    *)
Type
   EndpointRecord=Record X,Y,Z:integer; End;
   EndpointArrayType=Array[1..255] Of EndpointRecord;
   EndpointArrayPointer=^EndpointArrayType;
   LineRecord=Record Endpoint1,Endpoint2,Color:Byte; End;
   LineArrayType=Array[1..255] Of LineRecord;
   LineArrayPointer=^LineArrayType;
Procedure InitObject(EndpointArray:EndpointArrayPointer;
   LineArray:LineArrayPointer;
   Endpoints,Lines:Byte;
   ScaleFactor:Integer;

   Var WorldEndpointList:WorldEndpointPointer;
   Var WorldLineList:WorldLinePointer);
const
  FreeWorldEndpointList:WorldEndpointPointer=Nil;
```

```
  FreeWorldLineList:WorldLinePointer=Nil;
Var
  TempWorldEndpoint:WorldEndpointPointer;
  TempWorldLine:WorldLinePointer;
  TempEndpointPointers:Array[1..255] Of WorldEndpointPointer;
  I:Byte;
Begin

For I:=1 To Endpoints Do Begin
  If FreeWorldEndpointList=Nil Then New(TempWorldEndpoint)
  Else Begin TempWorldEndpoint:=FreeWorldEndpointList;
 FreeWorldEndpointList:=FreeWorldEndpointList^.Next; End;
  TempWorldEndpoint^.Transform3DEndpointVar.WorldX:=
ScaleFactor*EndpointArray^[I].X;
  TempWorldEndpoint^.Transform3DEndpointVar.WorldY:=
ScaleFactor*EndpointArray^[I].Y;
  TempWorldEndpoint^.Transform3DEndpointVar.WorldZ:=
ScaleFactor*EndpointArray^[I].Z;
  TempWorldEndpoint^.Next:=WorldEndpointList;
  TempEndpointPointers[I]:=TempWorldEndpoint;
  WorldEndpointList:=TempWorldEndpoint; End;
For I:=1 To Lines Do Begin
  If FreeWorldLineList=Nil Then New(TempWorldLine)
  Else Begin TempWorldLine:=FreeWorldLineList;
 FreeWorldLineList:=FreeWorldLineList^.Next; End;
  TempWorldLine^.Endpoint1:=TempEndpointPointers[LineArray^[I].Endpoint1];
  TempWorldLine^.Endpoint2:=TempEndpointPointers[LineArray^[I].Endpoint2];
  If LineArray^[I].Color=MaximumColor Then
    TempWorldLine^.Color:=LineArray^[I].Color
  Else TempWorldLine^.Color:=((LineArray^[I].Color-1) Mod MaximumColor)+1;
  TempWorldLine^.Next:=WorldLineList;
  WorldLineList:=TempWorldLine; End;
End;
(*----------------------------------------------------------------------*)


Procedure GenerateAxis(xmax,ymax,zmax:real;var xBarstar:mathmat.matx;
      var xhigh,xlow:realarraympbymp;
      var variance:mathmat.matx;evalue,eigenvalues:mathmat.matx);



Type
  EndPointNames=
  (dummy,
  end1,
  end2,
  end3,
  end4,
  end5,
  end6,
  end7,
```

```
  end8,
  end9,
  end10,
  end11,
  end12,
  end13,
  end14,
  end15,
  end16  (* centroid *)
  );

  EndPtArrayType=Array[1..16,1..3] of integer;
   CordLineRecord=Record Ep1,Ep2:EndpointNames; Color:Byte; End;
    CordLineArrayType=Array[1..13] Of CordLineRecord;
 const
 CordLineArray:CordLineArrayType=
 (
  (EP1:end1;Ep2:end2;color:white),
  (EP1:end1;Ep2:end3;color:white),
  (EP1:end1;Ep2:end4;color:white),
  (EP1:end1;Ep2:end5;color:white),
  (EP1:end1;Ep2:end6;color:white),
  (EP1:end1;Ep2:end7;color:white),
  (EP1:end8;Ep2:end14;color:cyan),
  (EP1:end9;Ep2:end14;color:cyan),
  (EP1:end10;Ep2:end14;color:cyan),
  (EP1:end11;Ep2:end14;color:cyan),
  (EP1:end12;Ep2:end14;color:cyan),
  (EP1:end13;Ep2:end14;color:cyan),
  (EP1:end15;EP2:end16;color:cyan));


Const

 Magnify:Real=150.1;  {Indicates relative length of Axes to object}

var
 CordEndPtArray:EndPtArrayType;
 DX,DY,DZ,gd,gm: Integer;    {Dummy Variables}
 MX,MY,MZ: Real;
 Center: Real;        {Center of Label}
 Leg: Real;          {Label variable}
 tempreal:real;
Begin

(* MX := Xmax*3.0; MY := Ymax*3.0; MZ := Zmax*3.0;
If Xmax<((MY+MZ)/3) then Xmax:=(MY+MZ)/3;
If Ymax<((MX+MZ)/3) then Ymax:=(MX+MZ)/3;
If Zmax<((MX+MY)/3) then Zmax:=(MX+MY)/3;  *)
```

```
Xmax:= Magnify*Xmax;
Ymax:= Magnify*Ymax;   {We want the Axes to go past the object}
Zmax:= Magnify*Zmax;
IF xmax
IF ymax

   CordEndPtArray[14,1]:=0;
   CordEndPtArray[14,2]:=0;  (* Centroid *)
   CordEndPtArray[14,3]:=0;


  CordEndPtArray[1,1]:=0-Round(100*xBarstar.data[1,1]);
   CordEndPtArray[1,2]:=0-Round(100*xBarstar.data[2,1]);  (* origin *)
   if ordp=3 then
   CordEndPtArray[1,3]:=0-Round(100*xBarstar.data[3,1])
   else
   CordEndPtArray[1,3]:=0;

  CordEndPtArray[2,1]:=0-Round(100*xBarstar.data[1,1]);
   CordEndPtArray[2,2]:=0-(Round(100*xBarstar.data[2,1]+ymax));
   If ordp=3 then
   CordEndPtArray[2,3]:=0-Round(100*xBarstar.data[3,1])
   else
   CordEndPtArray[2,3]:=0;

  CordEndPtArray[3,1]:=0-Round(100*xBarstar.data[1,1]);
   CordEndPtArray[3,2]:=0-Round(100*xBarstar.data[2,1]);
   if ordp=3 then
   CordEndPtArray[3,3]:=0-(Round(100*xBarstar.data[3,1]+zmax))
   else
   CordEndPtArray[3,3]:=0;


  CordEndPtArray[4,1]:=0-(Round(100*xBarstar.data[1,1]+xmax));
   CordEndPtArray[4,2]:=0-Round(100*xBarstar.data[2,1]);
   if ordp=3 then
   CordEndPtArray[4,3]:=0-Round(100*xBarstar.data[3,1])
   else
   CordEndPtArray[4,3]:=0;

  CordEndPtArray[5,1]:=0-Round(100*xBarstar.data[1,1]);
   CordEndPtArray[5,2]:=0-(Round(100*xBarstar.data[2,1]-ymax));
   if ordp=3 then
   CordEndPtArray[5,3]:=0-Round(100*xBarstar.data[3,1])
   else
   CordEndPtArray[5,3]:=0;

  CordEndPtArray[6,1]:=0-Round(100*xBarstar.data[1,1]);
   CordEndPtArray[6,2]:=0-Round(100*xBarstar.data[2,1]);
   if ordp=3 then
   CordEndPtArray[6,3]:=0-(Round(100*xBarstar.data[3,1]-zmax))
   else
```

```
 CordEndPtArray[6,3]:=0;

CordEndPtArray[7,1]:=0-(Round(100*xBarstar.data[1,1]-xmax));
 CordEndPtArray[7,2]:=0-Round(100*xBarstar.data[2,1]);
  if ordp=3 then
 CordEndPtArray[7,3]:=0-Round(100*xBarstar.data[3,1])
 else
 CordEndPtArray[7,3]:=0;

(* Endpoints at the Ends of the Major and Minor Axis *)

  If (xhigh[1,1]-xlow[1,1])>(xhigh[1,2]-xlow[1,2])
  Then begin

     ellipseX1X2bottom(xhigh[1,1],variance,cvalue,xBarstar,answer);
 CordEndPtArray[8,1]:=0-(Round(100*(xhigh[1,1]-xBarstar.data[1,1])));
 CordEndPtArray[8,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
  CordEndPtArray[8,3]:=0;

   ellipseX1X2top(xlow[1,1],variance,cvalue,xBarstar,answer);
 CordEndPtArray[9,1]:=0-(Round(100*(xlow[1,1]-xBarstar.data[1,1])));
 CordEndPtArray[9,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
 CordEndPtArray[9,3]:=0;

 If ordp=2 then begin
      tempreal:=xBarstar.data[1,1]-
           (eigenvalues.data[4,2]*cos(eigenvalues.data[1,2]));

      ellipseX1X2top(tempreal,variance,cvalue,xBarstar,answer);
 CordEndPtArray[15,1]:=0-(round(100*(tempreal-xbarstar.data[1,1])));
 CordEndPtArray[15,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
  CordEndPtArray[15,3]:=0;

    tempreal:=xBarstar.data[1,1]+
           (eigenvalues.data[4,2]*cos(eigenvalues.data[1,2]));
  ellipseX1X2bottom(xBarstar.data[1,1],variance,cvalue,xBarstar,answer);
 CordEndPtArray[16,1]:=0-(round(100*(-xbarstar.data[1,1]+tempreal)));
 CordEndPtArray[16,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
 CordEndPtArray[16,3]:=0;
   end;end

    else begin

  ellipseX2X1Bottom(xhigh[1,2],variance,cvalue,xBarstar,answer);
 CordEndPtArray[8,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
 CordEndPtArray[8,2]:=0-(Round(100*(xhigh[1,2]-xBarstar.data[2,1])));
 CordEndPtArray[8,3]:=0;

 ellipseX2X1top(xlow[1,2],variance,cvalue,xBarstar,answer);
 CordEndPtArray[9,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
 CordEndPtArray[9,2]:=0-(Round(100*(xlow[1,2]-xBarstar.data[2,1])));
 CordEndPtArray[9,3]:=0;
```

```
 If ordp=2 then begin
      tempreal:=xBarstar.data[2,1]-
            (eigenvalues.data[4,2]*cos(eigenvalues.data[1,2]));
 ellipseX2X1top(tempreal,variance,cvalue,xBarstar,answer);
CordEndPtArray[15,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
CordEndPtArray[15,2]:=0-(Round(100*(tempreal-xbarstar.data[2,1])));
 CordEndPtArray[15,3]:=0;
      tempreal:=xBarstar.data[2,1]+
            (eigenvalues.data[4,2]*cos(eigenvalues.data[1,2]));
  ellipseX2X1bottom(tempreal,variance,cvalue,xBarstar,answer);
CordEndPtArray[16,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
CordEndPtArray[16,2]:=0-(Round(100*(-xbarstar.data[2,1]+tempreal)));
CordEndPtArray[16,3]:=0;
  end;end;


If ordp=3 then  begin

   If (xhigh[1,1]-xlow[1,1])>(xhigh[1,3]-xlow[1,3])
  Then begin

 ellipseX1X3top(xhigh[1,1],variance,cvalue,xBarstar,answer);
 CordEndPtArray[10,1]:=0-(Round(100*(xhigh[1,1]-xBarstar.data[1,1])));
 CordEndPtArray[10,2]:=0;
 CordEndPtArray[10,3]:=0-(Round(100*(answer-xBarstar.data[3,1])));


  ellipseX1X3bottom(xlow[1,1],variance,cvalue,xBarstar,answer);
 CordEndPtArray[11,1]:=0-(Round(100*(xlow[1,1]-xBarstar.data[1,1])));
 CordEndPtArray[11,2]:=0;
 CordEndPtArray[11,3]:=0-(Round(100*(answer-xBarstar.data[3,1])));

 end else begin

 ellipseX3X1top(xhigh[1,3],variance,cvalue,xBarstar,answer);
 CordEndPtArray[10,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
 CordEndPtArray[10,2]:=0;
 CordEndPtArray[10,3]:=0-(Round(100*(xhigh[1,3]-xBarstar.data[3,1])));

  ellipseX3X1bottom(xlow[1,3],variance,cvalue,xBarstar,answer);
 CordEndPtArray[11,1]:=0-(Round(100*(answer-xBarstar.data[1,1])));
 CordEndPtArray[11,2]:=0;
 CordEndPtArray[11,3]:=0-(Round(100*(xlow[1,3]-xBarstar.data[3,1])));
 end;  {end if x1 or x3 major axis }

  If (xhigh[1,2]-xlow[1,2])>(xhigh[1,3]-xlow[1,3])
  Then begin

 ellipseX2X3top(xhigh[1,2],variance,cvalue,xBarstar,answer);
 CordEndPtArray[12,1]:=0;
 CordEndPtArray[12,2]:=0-(Round(100*(xhigh[1,2]-xBarstar.data[2,1])));
```

```
  CordEndPtArray[12,3]:=0-(Round(100*(answer-xBarstar.data[3,1])));

  ellipseX2X3bottom(xlow[1,2],variance,cvalue,xBarstar,answer);
  CordEndPtArray[13,1]:=0;
  CordEndPtArray[13,2]:=0-(Round(100*(xlow[1,2]-xBarstar.data[2,1])));
  CordEndPtArray[13,3]:=0-(Round(100*(answer-xBarstar.data[3,1])));
  end
  else begin

  ellipseX3X2top(xhigh[1,3],variance,cvalue,xBarstar,answer);
  CordEndPtArray[12,1]:=0;
  CordEndPtArray[12,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
  CordEndPtArray[12,3]:=0-(Round(100*(xhigh[1,3]-xBarstar.data[3,1])));

  ellipseX3X2bottom(xlow[1,3],variance,cvalue,xBarstar,answer);
  CordEndPtArray[13,1]:=0;
  CordEndPtArray[13,2]:=0-(Round(100*(answer-xBarstar.data[2,1])));
  CordEndPtArray[13,3]:=0-(Round(100*(xlow[1,3]-xBarstar.data[3,1])));
  end; { end if x2 or x3 major axis }

   end
  else  begin
  CordEndPtArray[10,1]:=0;
  CordEndPtArray[10,2]:=0;
  CordEndPtArray[10,3]:=0;

  CordEndPtArray[11,1]:=0;
  CordEndPtArray[11,2]:=0;
  CordEndPtArray[11,3]:=0;

  CordEndPtArray[12,1]:=0;
  CordEndPtArray[12,2]:=0;
  CordEndPtArray[12,3]:=0;

  CordEndPtArray[13,1]:=0;
  CordEndPtArray[13,2]:=0;
  CordEndPtArray[13,3]:=0;

  end;  (*  if ordp = 3 *)

if ordp=2 then
InitObject(@CordEndPtArray,@CordLineArray,16,13,1,
       WorldEndPointList,WorldLineList)
else
InitObject(@CordEndPtArray,@CordLineArray,14,12,1,
      WorldEndPointList,WorldLineList);
end; {Procedure GenerateAxis}


Procedure ellipses(txbar:mathmat.matx;ordchoice,ordp:integer;var xlow,xhigh:realarraymp-
bymp;
      var xBarstar,
```

```
      cvalue:mathmat.matx;var variance:mathmat.matx;
      Var WorldPointList:WorldPointPointer;kolor:byte;whichHiLo,screen:Integer);
var
nop:integer; {number of points}
nopr:real;    {# of intervals}
Begin
 if ordp=3 then  begin
 nopr:=150.0;
 nop:=151;end
 else begin
 nopr:=275.0;
 nop:=276;end;
 if ordchoice=5 then begin
 nopr:=130.0;
 nop:=131; end;
(*------------------------------------------------------------------------*)
(* Calculate coordinates for points in the X1 vs X2 Ellipse.           *)
(*------------------------------------------------------------------------*)
 If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])
    Then begin
      xx:=xlow[whichHiLo,1];
       change:=(xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])/nopr;end
       else begin
       xx:=xlow[whichHiLo,2];
       change:=(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])/nopr;end;


For J:=1 to nop  Do Begin

  New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
       WorldPointList:=TempWorldPoint;

  If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])
    Then begin
  ellipseX1X2Top(xx,Variance,Cvalue,xBarstar,answer);

  WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (xx-xBarstar.data[1,1]));
  WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (answer-xBarstar.data[2,1]));
  WorldPointList^.Transform3DPointVar.WorldZ:=0; {zbar} end
  else begin
   ellipseX2X1Top(xx,Variance,Cvalue,xBarstar,answer);

  WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (answer-xBarstar.data[1,1]));
  WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (xx-xBarstar.data[2,1]));
  WorldPointList^.Transform3DPointVar.WorldZ:=0; {zbar} end;

  WorldPointList^.Color:=kolor;
       xx:=xx+change;
       End;

(* The Points for The Bottom of Ellipse 1*)
     If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])
```

```
  Then begin
    xx:=xlow[whichHiLo,1];
     change:=(xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])/nopr;end
     else begin
     xx:=xlow[whichHiLo,2];
     change:=(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])/nopr;end;

for j:=1 to nop do Begin
 New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
      WorldPointList:=TempWorldPoint;
    (* This if looks to see which variable represents the major axis *)
    (*  Then graphs using it as xx *)

    If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])
  Then begin
 ellipseX1X2Bottom(xx,Variance,Cvalue,xBarstar,answer);

 WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (xx-xBarstar.data[1,1]));
 WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (answer-xBarstar.data[2,1]));
 WorldPointList^.Transform3DPointVar.WorldZ:=0;  {zbar} end
 else begin
  ellipseX2X1Bottom(xx,Variance,Cvalue,xBarstar,answer);

 WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (answer-xBarstar.data[1,1]));
 WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (xx-xBarstar.data[2,1]));
 WorldPointList^.Transform3DPointVar.WorldZ:=0;  {zbar} end;

       WorldPointList^.Color:=kolor;
       xx:=xx+change;
       End;

If (ordp=3) then begin

 (*  Calculate the X1 vs X3 Ellipse *)
   If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,3]-xlow[WhichHiLo,3])
   Then begin
       xx:=xlow[whichHiLo,1];
       change:=(xhigh[whichHiLo,1]-xlow[whichHiLo,1])/nopr;end
       else begin
        xx:=xlow[whichHiLo,3];
       change:=(xhigh[whichHiLo,3]-xlow[whichHiLo,3])/nopr;end;

 For j:=1 to nop do begin


       New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
         WorldPointList:=TempWorldPoint;

       If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,3]-xlow[Which-
HiLo,3])
   Then begin
       ellipseX1X3top(xx,Variance,Cvalue,xBarstar,answer);
```

```
        WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (xx-xBar-
star.data[1,1]));
        WorldPointList^.Transform3DPointVar.WorldY:= 0;  {Ybar}

        WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (answer-xBar-
star.data[3,1]));
        end
        else begin
        ellipseX3X1top(xx,Variance,Cvalue,xBarstar,answer);

        WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (answer-xBar-
star.data[1,1]));
        WorldPointList^.Transform3DPointVar.WorldY:= 0;  {Ybar}

        WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (xx-xBar-
star.data[3,1]));
        end;


        WorldPointList^.Color:=kolor;
        xx:=xx+change;
        End;

      If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,3]-xlow[Which-
HiLo,3])
   Then begin
        xx:=xlow[whichHiLo,1];
        change:=(xhigh[whichHiLo,1]-xlow[whichHiLo,1])/nopr;end
        else begin
         xx:=xlow[whichHiLo 3];
        change:=(xhigh[whichHiLo,3]-xlow[whichHiLo,3])/nopr;end;

 For j:=1 to nop do begin


      New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
        WorldPointList:=TempWorldPoint;

      If (xhigh[WhichHiLo,1]-xlow[WhichHiLo,1])>(xhigh[WhichHiLo,3]-xlow[Which-
HiLo,3])
   Then begin
        ellipseX1X3bottom(xx,Variance,Cvalue,xBarstar,answer);

        WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (xx-xBar-
star.data[1,1]));
        WorldPointList^.Transform3DPointVar.WorldY:= 0;  {Ybar}

        WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (answer-xBar-
star.data[3,1]));
        end
        else begin
```

```
        ellipseX3X1bottom(xx,Variance,Cvalue,xBarstar,answer);

        WorldPointList^.Transform3DPointVar.WorldX:= Round( 100* (answer-xBar-
star.data[1,1]));
        WorldPointList^.Transform3DPointVar.WorldY:= 0;  {Ybar}

        WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (xx-xBar-
star.data[3,1]));
        end;
         WorldPointList^.Color:=kolor;
        xx:=xx+change;
        End;

(* This section calculates the points in the X2 vs X3 ellipse *)

  If (xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])>(xhigh[WhichHiLo,3]-xlow[WhichHiLo,3])
   Then begin
    xx:=xlow[whichHiLo,2];
    change:=(xhigh[whichHiLo,2]-xlow[whichHiLo,2])/nopr; end
    else begin
     xx:=xlow[whichHiLo,3];
    change:=(xhigh[whichHiLo,3]-xlow[whichHiLo,3])/nopr; end;

For J:=1 to nop do begin
      New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
      WorldPointList:=TempWorldPoint;

        If (xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])>(xhigh[WhichHiLo,3]-xlow[Which-
HiLo,3])
   Then begin
    ellipseX2X3top(xx,Variance,Cvalue,xBarstar,answer);
    WorldPointList^.Transform3DPointVar.WorldX:= 0;  {xBarstar}
    WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (xx-xBarstar.data[2,1]));
    WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (answer-xBar-
star.data[3,1]));
     end
     else begin

     ellipseX3X2top(xx,Variance,Cvalue,xBarstar,answer);
    WorldPointList^.Transform3DPointVar.WorldX:= 0;  {xBarstar}
    WorldPointList^.Transform3DPointVar.WorldY:= Round( 100* (answer-xBar-
star.data[2,1]));
    WorldPointList^.Transform3DPointVar.WorldZ:= Round( 100* (xx-xBarstar.data[3,1]));
     end;
    WorldPointList^.Color:=kolor;
    xx:=xx+change;
    End;


    If (xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])>(xhigh[WhichHiLo,3]-xlow[WhichHiLo,3])
   Then begin
    xx:=xlow[whichHiLo,2];
```

```
    change:=(xhigh[whichHiLo,2]-xlow[whichHiLo,2])/nopr; end
    else begin
     xx:=xlow[whichHiLo,3];
    change:=(xhigh[whichHiLo,3]-xlow[whichHiLo,3])/nopr; end;

For J:=1 to nop do begin
      New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
      WorldPointList:=TempWorldPoint;

        If (xhigh[WhichHiLo,2]-xlow[WhichHiLo,2])>(xhigh[WhichHiLo,3]-xlow[Which-
HiLo,3])
   Then begin
    ellipseX2X3bottom(xx,Variance,Cvalue,xBarstar,answer);
    WorldPointList^.Transform3DPointVar.WorldX:=  0;   {xBarstar}
    WorldPointList^.Transform3DPointVar.WorldY:=  Round( 100* (xx-xBarstar.data[2,1]));
    WorldPointList^.Transform3DPointVar.WorldZ:=  Round( 100* (answer-xBar-
star.data[3,1]));
     end
     else begin

     ellipseX3X2bottom(xx,Variance,Cvalue,xBarstar,answer);
    WorldPointList^.Transform3DPointVar.WorldX:=  0;   {xBarstar}
    WorldPointList^.Transform3DPointVar.WorldY:=  Round( 100* (answer-xBar-
star.data[2,1]));
    WorldPointList^.Transform3DPointVar.WorldZ:=  Round( 100* (xx-xBarstar.data[3,1]));
     end;
    WorldPointList^.Color:=kolor;
    xx:=xx+change;
    End;
end; (*if*)

   New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
     WorldPointList:=TempWorldPoint;
   (* Plot Centroid *)
   (* Note centroid is moved to origin so it is the point of rotation *)
    WorldPointList^.Transform3DPointVar.WorldX:=0;

   WorldPointList^.Transform3DPointVar.WorldY:=0;

   WorldPointList^.Transform3DPointVar.WorldZ:=0;
      WorldPointList^.Color:=white;

  If (ordchoice=4)  and (screen=3)  then begin

    New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
     WorldPointList:=TempWorldPoint;
   WorldPointList^.Transform3DPointVar.WorldX:=
           Round( 100* (MU.data[1,1]-xBarstar.data[1,1]));
   WorldPointList^.Transform3DPointVar.WorldY:=
           Round( 100* (Mu.data[2,1]-xBarstar.data[2,1]));
   if ordp=3 then
   WorldPointList^.Transform3DPointVar.WorldZ:=
```

```
        Round( 100* (Mu.data[3,1]-xBarstar.data[3,1]))
        else
        WorldPointList^.Transform3DPointVar.WorldZ:=0;
  WorldPointList^.Color:=white; end;

 If (ordchoice=5) and (screen=3)  then begin

   New(TempWorldPoint);TempWorldPoint^.Next:=Worldpointlist;
    WorldPointList:=TempWorldPoint;
  WorldPointList^.Transform3DPointVar.WorldX:=
        Round( 100* (txbar.data[1,1]-xBarstar.data[1,1]));
  WorldPointList^.Transform3DPointVar.WorldY:=
        Round( 100* (txbar.data[2,1]-xBarstar.data[2,1]));
  if ordp=3 then
  WorldPointList^.Transform3DPointVar.WorldZ:=
        Round( 100* (txbar.data[3,1]-xBarstar.data[3,1]))
        else
        WorldPointList^.Transform3DPointVar.WorldZ:=0;
  WorldPointList^.Color:=white; end;

 End; (* Procedure Ellipses*)

 Procedure DrawOutlines;
(* Draw lines for the boundaries around the windows.  Make sure to draw the *)
(* lines in all of the buffers that we'll be using (2 buffers at most).    *)
Begin
For I:=MaximumBuffer Downto 0 Do Begin
  SetDrawingBufferVar.Buffer:=I;
 (*  This first section draws the lines and headings that
 are common to all screens *)SetActivePage(I);
  SetViewPort(1,1,WindowVertical,WindowHorizontal,true);
  ClearViewPort;
  AcroMole.SetDrawingBuffer(SetDrawingBufferVar);
  DrawLineVar.Color:=MaximumColor;
  DrawLineVar.ScreenX1:=MinimumScreenX;
  DrawLineVar.ScreenY1:=MinimumScreenY;
  DrawLineVar.ScreenX2:=MinimumScreenX;
  DrawLineVar.ScreenY2:=MaximumScreenY;
  AcroMole.DrawLine(DrawLineVar);
  DrawLineVar.ScreenX1:=MaximumScreenX;
  DrawLineVar.ScreenY1:=MaximumScreenY;
  AcroMole.DrawLine(DrawLineVar);
  DrawLineVar.ScreenX2:=MaximumScreenX;
  DrawLineVar.ScreenY2:=MinimumScreenY;
  AcroMole.DrawLine(DrawLineVar);
  DrawLineVar.ScreenX1:=MinimumScreenX;
  DrawLineVar.ScreenY1:=MinimumScreenY;
  AcroMole.DrawLine(DrawLineVar);
  DrawLineVar.ScreenX1:=WindowVertical;
  DrawLineVar.ScreenX2:=WindowVertical;
  DrawLineVar.ScreenY2:=MaximumScreenY;
  AcroMole.DrawLine(DrawLineVar);
```

```
DrawLineVar.ScreenY1:=WindowHorizontal;
DrawLineVar.ScreenX2:=MaximumScreenX;
DrawLineVar.ScreenY2:=WindowHorizontal;
AcroMole.DrawLine(DrawLineVar);
DrawLineVar.ScreenX1:=MinimumScreenX;
DrawLineVar.ScreenY1:=WindowHorizontal;
DrawLineVar.ScreenX2:=WindowVertical;
DrawLineVar.ScreenY2:=WindowHorizontal;
AcroMole.DrawLine(DrawLineVar);

DrawLineVar.ScreenX1:=MinimumScreenX;
DrawLineVar.ScreenY1:=WindowHorizontal-12;
DrawLineVar.ScreenX2:=MaximumScreenX;
DrawLineVar.ScreenY2:=WindowHorizontal-12;
AcroMole.DrawLine(DrawLineVar);

DrawLineVar.ScreenX1:=WindowVertical;
DrawLineVar.ScreenY1:=MaximumScreenY-12;
DrawLineVar.ScreenX2:=MaximumScreenX;
DrawLineVar.ScreenY2:=MaximumScreenY-12;
AcroMole.DrawLine(DrawLineVar);


 End;



For I:=MaximumBuffer Downto 0 Do Begin
  setactivepage(I);
  SetTextJustify(0,1);
  SetColor(white);
Case Ordchoice of
1: Begin
   Gmatwrite(xbar,4,2,#230,15,5,lightgray,blue,white);
 Gmatwrite(Sigma,5,2,#228,138,5,lightgray,blue,white);
 Gmatwrite(SigmabyN,5,2,#228'/n',130,45,lightgray,blue,white);
 Gmatwrite(eigenvalues,5,2,'Lambda',30,85,lightgray,blue,white);
 Gmatwrite(Eigenvectors,5,2,'EV',149,85,lightgray,blue,white);
 GmatWrite(evalstar,5,2,'L.*',20,125,lightgray,blue,white);
 GmatWrite(evecstar,5,2,'EV.*',140,125,lightgray,blue,white);
  SetTextJustify(2,1);

 OutTextXy(635,(maximumscreeny div 2+170),'This Screen Intentionally Blank');
 OutTextXY(200,(maximumscreeny div 2+95),'Screen 2 - '#230', '#228);
 OutTextXy(550,(maximumscreeny div 2+95),'Screen 3');
 OutTextXY(550,7,'Screen 1 - '#230', '#228'/N');


end; {choice 1}
2,3: Begin
     Gmatwrite(xbar,4,2,'Xbar',25,5,lightgray,blue,white);
Gmatwrite(S,5,2,'S',138,5,lightgray,blue,white);
```

```
Gmatwrite(SbyN,5,2,'S/n',130,45,lightgray,blue,white);
Gmatwrite(eigenvalues,5,2,'Lambda',30,85,lightgray,blue,white);
Gmatwrite(Eigenvectors,5,2,'EV',149,85,lightgray,blue,white);
GmatWrite(evalstar,5,2,'L.*',20,125,lightgray,blue,white);
GmatWrite(evecstar,5,2,'EV.*',140,125,lightgray,blue,white);
 SetTextJustify(2,1);

OutTextXY(200,(maximumscreeny div 2+95),'Screen 2 - Xbar, S');
OutTextXy(550,(maximumscreeny div 2+95),'Screen 3');
OutTextXY(550,7,'Screen 1 - Xbar, S/N'); end;

4: Begin
Gmatwrite(xbar,4,2,'Xbar',25,5,lightgray,blue,white);
Gmatwrite(Mu,5,2,#230,138,5,lightgray,blue,white);
GmatWrite(sbyn,5,2,'S/n',140,45,lightgray,blue,white);
Gmatwrite(eigenvalues,5,2,'Lambda',30,85,lightgray,blue,white);
Gmatwrite(Eigenvectors,5,2,'EV',149,85,lightgray,blue,white);
GmatWrite(evalstar,5,2,'L.*',20,125,lightgray,blue,white);
GmatWrite(evecstar,5,2,'EV.*',140,125,lightgray,blue,white);
 SetTextJustify(2,1);
  OutTextXY(200,(maximumscreeny div 2+95),'Screen 2 - S');
OutTextXy(550,(maximumscreeny div 2+95),'Screen 3 - Test of Mean');
OutTextXY(550,7,'Screen 1 - S/n'); end;

5: Begin
Gmatwrite(xbar,4,2,#230'.*',25,5,lightgray,blue,white);
Gmatwrite(Mu,5,2,#230,138,5,lightgray,blue,white);
Gmatwrite(txbar,4,2,#230'.0',25,45,lightgray,blue,white);
GmatWrite(Sigma,5,2,#228,143,45,lightgray,blue,white);
Gmatwrite(eigenvalues,5,2,'Lambda',30,85,lightgray,blue,white);
Gmatwrite(Eigenvectors,5,2,'EV',149,85,lightgray,blue,white);
GmatWrite(evalstar,5,2,'L.*',20,125,lightgray,blue,white);
GmatWrite(S,5,2,'S',140,125,lightgray,blue,white);
 SetTextJustify(2,1);
  OutTextXY(200,(maximumscreeny div 2+95),'Screen 2 - '#228'(Red) & S');
OutTextXy(570,(maximumscreeny div 2+95),'Screen 3 - Test of Mean - S/n');
OutTextXY(550,7,'Screen 1 - '#228'/n'); end;
end; {case}

 End;{for}

SetDrawingBufferVar.Buffer:=MaximumBuffer;
AcroMole.SetDrawingBuffer(SetDrawingBufferVar);
end; {outlines}

 (* BEGIN Main Program *)
Begin

OvrInit('a3scrloo.OVR');
OvrInitEMS;
```

```
888:
StartAcroMole;
 i:=1;
  Gd:=3; Gm:=EGAHi;
   InitGraph(Gd,Gm,'');
   errcode:=GraphResult;
   If Graphresult <> grOK  then begin
   Writeln('Graphics Error  ',GraphErrorMsg(graphresult));
   readln; Halt(1);  end;
    SetBkColor(1); {Background Blue}
     TextColor(1);
     (* set some flags *)
     flagf1:=false;
     flagf2:=false;
     Flagf3:=false;

Repeat
OutTextXY(30,120,'Do you wish to deal with the Bivariate Or MultiVariate Normal Distribu-
tion?');
OutTextXY(35,145,'Input 2 for BVN (2-D) or 3 for MVN (3-D):');
repeat
p:=ReadKey;
val(P,ordP,code);
until (ordp=2) or (ordp=3);
clrscr;
val(P,ordP,code);

Case OrdP of
2: begin
  OutTextXY(70,120,'This is the Case of a Bivariate Normal Distribution:');
OutTextXY(90,135,'(1) Study the Theoretical Graphs');
OutTextXY(90,150,'(2) Study Graphs for Empirical Data Read from Disk (a:\empdata.dat)');
OutTextXY(90,165,'(3) Study Graphs from Empirical Data from Keyboard');
OutTextXY(90,180,'(4) Read Data from a file on disk (Test of Mean Vector)');
OutTExtXY(90,195,'(5) Theoretical & Empirical Graphs + Test of Mean');
OutTextXY(80,210,'Type the number of choice (1, 2, 3, 4 or 5)');
Repeat
choice:=Readkey;
val(choice,ordchoice,code);
until (ordchoice=1) or (ordchoice=2) or (ordchoice=3) or
    (ordchoice=4) or (ordchoice=5);
clrscr; End; {case 2}

3: begin
OutTextXY(70,120,'This is the Case of a Multivariate Normal Distribution:');
OutTextXY(90,135,'(1) Study the Theoretical Graphs');
OutTextXY(90,150,'(2) Study Graphs for Empirical Data Read from Disk (a:\empdata.dat)');
OutTextXY(90,165,'(3) Study Graphs from Empirical Data from Keyboard');
OutTextXY(90,180,'(4) Read Data from a file on disk (Test of Mean Vector)');
OutTExtXY(90,195,'(5) Theoretical & Empirical Graphs + Test of Mean');
OutTextXY(80,210,'Type the number of choice (1, 2, 3, 4 or 5)');
Repeat
```

```
choice:=Readkey;
val(choice,ordchoice,code);
until (ordchoice=1) or (ordchoice=2) or (ordchoice=3) or
   (ordchoice=4) or (ordchoice=5);

clrscr; End; {case 3}

end; {case ordp}

 look:=false;
Case Ordchoice of

   1: GenerateData(txbar,xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SigmabyN,evalstar,evecstar,ordp,sigma,sbyn);
   2: ReadDataEmpirical(xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SbyN,evalstar,evecstar,ordp);
   3: GetDataKeyboard(xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SbyN,evalstar,evecstar,ordp);
   4: ReadDataTest(xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SbyN,evalstar,evecstar,ordp);
   5: begin look:=true;
   GenerateData(txbar,xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SigmabyN,evalstar,evecstar,ordp,sigma,sbyn);
     end; {begin}
   Else
  GenerateData(txbar,xdata,S,xhigh,xlow,cvalue,mu,xbar,look,
     Eigenvalues,Eigenvectors,SigmabyN,evalstar,evecstar,ordp,sigma,sbyn);

   End; {Case}


(* This Section determines what MND is being looked at:
  If Ordchoice = 1 then Theoretical
  If OrdChoice = 2 or 3 then Empirical
  If Ordchoice = 4 then Testing  *)

Case Ordchoice of

   1: Begin
     WorldPointListS3:=nil;
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Mu,cvalue,Sigma,WorldPointLists2,2,1,2);
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Mu,cvalue,SigmabyN,WorldPointLists1,3,2,1);
     end;
   2,3: Begin
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,SbyN,WorldPointLists1,3,2,1);
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,S,WorldPointLists2,2,1,2);
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,SbyN,WorldPointLists3,2,2,4);
     end;

    4: Begin
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,SbyN,WorldPointLists1,3,2,1);
     ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,SbyN,WorldPointLists3,2,2,3);
```

```
    ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,S,WorldPointLists2,2,1,2);
    end;
  5: Begin
    ellipses(txbar,ordchoice,ordp,xlow,xhigh,Mu,cvalue,Sigma,WorldPointLists2,4,1,2);
    ellipses(txbar,ordchoice,ordp,xlow,xhigh,Mu,cvalue,SigmabyN,WorldPointLists1,4,2,1);
    ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,S,WorldPointLists2,3,3,2);
    ellipses(txbar,ordchoice,ordp,xlow,xhigh,Xbar,cvalue,Sbyn,WorldPointLists3,3,4,3);
    end;
  end; (* case *)


(*------------------------------------------------------------------------*)
xmax:=xhigh[1,1]; ymax:=xhigh[1,2]; zmax:=xhigh[1,3];
if (ordchoice=1) OR (ordchoice=5) then
 GenerateAxis(20.0,20.0,20.0,Mu,xhigh,xlow,Sigma,cvalue,eigenvalues)
 else
  GenerateAxis(20.0,20.0,20.0,xbar,xhigh,xlow,S,cvalue,eigenvalues);


(* This is the top of the main loop. *)

CalculateScaleFactorsVar.ResolutionX:=ResolutionX;
CalculateScaleFactorsVar.ResolutionY:=ResolutionY;
CalculateScaleFactorsVar.SizeX:=4;
CalculateScaleFactorsVar.SizeY:=3;
CalculateScaleFactorsVar.ScaleFactorLo:=0;
CalculateScaleFactorsVar.ScaleFactorHi:=550;
AcroMole.CalculateScaleFactors(CalculateScaleFactorsVar);

SetRight3DCameraVar.ScaleFactorX:=CalculateScaleFactorsVar.ScaleFactorX;
SetRight3DCameraVar.ScaleFactorY:=CalculateScaleFactorsVar.ScaleFactorY;
SetRight3DCameraVar.Perspective:=1000;

CalculateScaleFactorsVar.ResolutionX:=ResolutionX;
CalculateScaleFactorsVar.ResolutionY:=ResolutionY;
CalculateScaleFactorsVar.SizeX:=4;
CalculateScaleFactorsVar.SizeY:=3;
CalculateScaleFactorsVar.ScaleFactorLo:=0;
CalculateScaleFactorsVar.ScaleFactorHi:=550;
AcroMole.CalculateScaleFactors(CalculateScaleFactorsVar);

SetLeft3DCameraVar.ScaleFactorX:=CalculateScaleFactorsVar.ScaleFactorX;
SetLeft3DCameraVar.ScaleFactorY:=CalculateScaleFactorsVar.ScaleFactorY;
SetLeft3DCameraVar.Perspective:=1000;

WindowHorizontal:=(MaximumScreenY+MinimumScreenY) Div 2;
WindowVertical:=(MinimumScreenX+2*MaximumScreenX) Div 22;

(* Calculate the position of the left hand window. *)

WindowWidth:=WindowVertical-MinimumScreenX-1;
```

```
WindowHeight:=MaximumScreenY-MinimumScreenY-1;
SetLeftWindowVar.MinimumFilmX:=-(WindowWidth Div 2);
SetLeftWindowVar.MaximumFilmX:=SetLeftWindowVar.MinimumFilmX+WindowWidth-1;
SetLeftWindowVar.MinimumFilmY:=-(WindowHeight Div 2)+80;
SetLeftWindowVar.MaximumFilmY:=SetLeftWindowVar.MinimumFilmY+(WindowHeight
div 2)-13;
SetLeftWindowVar.MinimumScreenX:=MinimumScreenX+1;
SetLeftWindowVar.MinimumScreenY:=MinimumScreenY+1;

(* Calculate the position of the upper right window. *)

WindowWidth:=MaximumScreenX-WindowVertical-1;
WindowHeight:=MaximumScreenY-WindowHorizontal-1;
SetUpperWindowVar.MinimumFilmX:=-(WindowWidth Div 2);
SetUpperWindowVar.MaximumFilmX:=SetUpperWindowVar.MinimumFilmX+Window-
Width-1;
SetUpperWindowVar.MinimumFilmY:=-(WindowHeight Div 2);
SetUpperWindowVar.MaximumFilmY:=SetUpperWindowVar.MinimumFilmY+Window-
Height-13;
SetUpperWindowVar.MinimumScreenX:=WindowVertical+1;
SetUpperWindowVar.MinimumScreenY:=WindowHorizontal+1;


(* Calculate the position of the lower right window. *)

WindowHeight:=WindowHorizontal-MinimumScreenY-1;
SetLowerWindowVar.MinimumFilmX:=-(WindowWidth Div 2);
SetLowerWindowVar.MaximumFilmX:=SetLowerWindowVar.MinimumFilmX+Window-
Width-1;
SetLowerWindowVar.MinimumFilmY:=-(WindowHeight Div 2);
SetLowerWindowVar.MaximumFilmY:=SetLowerWindowVar.MinimumFilmY+Window-
Height-13;
SetLowerWindowVar.MinimumScreenX:=WindowVertical+1;
SetLowerWindowVar.MinimumScreenY:=MinimumScreenY+1;

SetFullWindowVar.MinimumFilmX:=MinimumScreenX+1;
SetFullWindowVar.MaximumFilmX:=MaximumscreenX-1;
SetFullWindowVar.MinimumFilmY:=MinimumScreenY+1;
SetFullWindowVar.MaximumFilmY:=MaximumScreenY-1;
SetFullWindowVar.MinimumScreenX:=MinimumScreenX+1;
SetFullWindowVar.MinimumScreenY:=MinimumScreenY+1;

DrawOutlines;

Repeat

  (* Set the clipping window for the left hand window *)

  AcroMole.SetWindow(SetLeftWindowVar);

  (* Set the camera position for a front view. *)
```

```
SetLeft3DCameraVar.WorldX:=-Round(Distance*Matrix[0,2]);
SetLeft3DCameraVar.WorldY:=-Round(Distance*Matrix[1,2]);
SetLeft3DCameraVar.WorldZ:=-Round(Distance*Matrix[2,2]);
SetLeft3DCameraVar.DirectionX:=Round(Matrix[0,2]);
SetLeft3DCameraVar.DirectionY:=Round(Matrix[1,2]);
SetLeft3DCameraVar.DirectionZ:=Round(Matrix[2,2]);
SetLeft3DCameraVar.UpX:=Round(Matrix[0,1]);
SetLeft3DCameraVar.UpY:=Round(Matrix[1,1]);
SetLeft3DCameraVar.UpZ:=Round(Matrix[2,1]);
AcroMole.Set3DCamera(SetLeft3DCameraVar);

(* Transform and clip the points. *)

TransformAndClip(WorldPointLists2,WorldEndPointList,WorldLineList);

(* Set the clipping window for the upper right window. *)

AcroMole.SetWindow(SetUpperWindowVar);

(* Set the camera position for a top view. *)


(* Transform and clip the points. *)
TransformAndClip(WorldPointListS1,WorldEndPointList,WorldLineList);


(* Set the clipping window for the lower right window. *)

AcroMole.SetWindow(SetLowerWindowVar);

(* Set the camera position for a right side view. *)

(* Transform and clip the points. *)
If ordchoice= 1 then
TransformAndClip(nil,nil,nil)
else
TransformAndClip(WorldPointLists3,WorldEndPointList,WorldLineList);

(* Draw the transformed points on the screen. *)

UpdateScreen;

(* Get the keyboard status and the change in time. *)

GetStatus;

(* Based on the keys pressed down, and the change *)
(* in time, update the angles and distance.    *)

Angle:=Speed*DeltaTime; SinPositiveAngle:=Sin(Angle); CosAngle:=Cos(Angle);

DeltaDistance:=0.5*Speed*DeltaTime;
```

```
If RightArrowFlag Xor LeftArrowFlag Then Begin
  If RightArrowFlag Then SinAngle:=SinPositiveAngle
  Else SinAngle:=-SinPositiveAngle;
  TempReal:=Matrix[0,0];
  Matrix[0,0]:=CosAngle*Matrix[0,0]-SinAngle*Matrix[0,2];
  Matrix[0,2]:=SinAngle*TempReal+CosAngle*Matrix[0,2];
  TempReal:=Matrix[1,0];
  Matrix[1,0]:=CosAngle*Matrix[1,0]-SinAngle*Matrix[1,2];
  Matrix[1,2]:=SinAngle*TempReal+CosAngle*Matrix[1,2];
  TempReal:=Matrix[2,0];
  Matrix[2,0]:=CosAngle*Matrix[2,0]-SinAngle*Matrix[2,2];
  Matrix[2,2]:=SinAngle*TempReal+CosAngle*Matrix[2,2]; End;

If UpArrowFlag Xor DownArrowFlag Then Begin
  If UpArrowFlag Then SinAngle:=SinPositiveAngle
  Else SinAngle:=-SinPositiveAngle;
  TempReal:=Matrix[0,1];
  Matrix[0,1]:=CosAngle*Matrix[0,1]-SinAngle*Matrix[0,2];
  Matrix[0,2]:=SinAngle*TempReal+CosAngle*Matrix[0,2];
  TempReal:=Matrix[1,1];
  Matrix[1,1]:=CosAngle*Matrix[1,1]-SinAngle*Matrix[1,2];
  Matrix[1,2]:=SinAngle*TempReal+CosAngle*Matrix[1,2];
  TempReal:=Matrix[2,1];
  Matrix[2,1]:=CosAngle*Matrix[2,1]-SinAngle*Matrix[2,2];
  Matrix[2,2]:=SinAngle*TempReal+CosAngle*Matrix[2,2]; End;

If PgUpFlag Xor PgDnFlag Then Begin
  If PgUpFlag Then SinAngle:=SinPositiveAngle
  Else SinAngle:=-SinPositiveAngle;
  TempReal:=Matrix[0,0];
  Matrix[0,0]:=CosAngle*Matrix[0,0]-SinAngle*Matrix[0,1];
  Matrix[0,1]:=SinAngle*TempReal+CosAngle*Matrix[0,1];
  TempReal:=Matrix[1,0];
  Matrix[1,0]:=CosAngle*Matrix[1,0]-SinAngle*Matrix[1,1];
  Matrix[1,1]:=SinAngle*TempReal+CosAngle*Matrix[1,1];
  TempReal:=Matrix[2,0];
  Matrix[2,0]:=CosAngle*Matrix[2,0]-SinAngle*Matrix[2,1];
  Matrix[2,1]:=SinAngle*TempReal+CosAngle*Matrix[2,1]; End;


(* This section checks to see if F1, F2 Or F3 has been pressed
  If so screen 1, 2 or 3 is inlarged and displayed as the
  only screen.  ESC returns all previous screens *)

If F1Flag or F2Flag or F3Flag
Then begin  (* display one large screen *)
Flagf1:=f1flag;
Flagf2:=f2flag;
Flagf3:=f3flag;
 (* First Clear Screen in both buffers *)
 For I:=MaximumBuffer Downto 0 Do Begin
```

```
  SetDrawingBufferVar.Buffer:=1;
  AcroMole.SetDrawingBuffer(SetDrawingBufferVar);
DrawRectangleVar.ScreenX1:=MinimumScreenX;
DrawRectangleVar.ScreenY1:=MinimumScreenY;
DrawRectangleVar.ScreenX2:=MaximumScreenX;
DrawRectangleVar.ScreenY2:=MaximumScreenY;
DrawRectangleVar.color:=blue;AcroMole.DrawRectangle(DrawRectangleVar);
end;
   (* Now display and rotate single screen *)

Repeat
 (* Draw the transformed points on the screen. *)

  AcroMole.SetWindow(SetFullWindowVar);
(* Set the camera position for a Full view. *)


 SetLeft3DCameraVar.WorldX:=-Round(Distance*Matrix[0,2]);
SetLeft3DCameraVar.WorldY:=-Round(Distance*Matrix[1,2]);
SetLeft3DCameraVar.WorldZ:=-Round(Distance*Matrix[2,2]);
SetLeft3DCameraVar.DirectionX:=Round(Matrix[0,2]);
SetLeft3DCameraVar.DirectionY:=Round(Matrix[1,2]);
SetLeft3DCameraVar.DirectionZ:=Round(Matrix[2,2]);
SetLeft3DCameraVar.UpX:=Round(Matrix[0,1]);
SetLeft3DCameraVar.UpY:=Round(Matrix[1,1]);
SetLeft3DCameraVar.UpZ:=Round(Matrix[2,1]);
AcroMole.Set3DCamera(SetLeft3DCameraVar);


(* Transform and clip the points. *)
 If flagf1 then
 TransformAndClip(WorldPointLists1,WorldEndPointList,WorldLineList);
 If FlagF2 then
 TransformAndClip(WorldPointLists2,WorldEndPointList,WorldLineList);
 If FlagF3 then
 TransformAndClip(WorldPointLists3,WorldEndPointList,WorldLineList);


UpdateScreen;

(* Get the keyboard status and the change in time. *)

GetStatus;

(* Based on the keys pressed down, and the change *)
(* in time, update the angles and distance.    *)

Angle:=Speed*DeltaTime; SinPositiveAngle:=Sin(Angle); CosAngle:=Cos(Angle);

DeltaDistance:=0.5*Speed*DeltaTime;

If RightArrowFlag Xor LeftArrowFlag Then Begin
```

```
    If RightArrowFlag Then SinAngle:=SinPositiveAngle
    Else SinAngle:=-SinPositiveAngle;
    TempReal:=Matrix[0,0];
    Matrix[0,0]:=CosAngle*Matrix[0,0]-SinAngle*Matrix[0,2];
    Matrix[0,2]:=SinAngle*TempReal+CosAngle*Matrix[0,2];
    TempReal:=Matrix[1,0];
    Matrix[1,0]:=CosAngle*Matrix[1,0]-SinAngle*Matrix[1,2];
    Matrix[1,2]:=SinAngle*TempReal+CosAngle*Matrix[1,2];
    TempReal:=Matrix[2,0];
    Matrix[2,0]:=CosAngle*Matrix[2,0]-SinAngle*Matrix[2,2];
    Matrix[2,2]:=SinAngle*TempReal+CosAngle*Matrix[2,2]; End;

  If UpArrowFlag Xor DownArrowFlag Then Begin
    If UpArrowFlag Then SinAngle:=SinPositiveAngle
    Else SinAngle:=-SinPositiveAngle;
    TempReal:=Matrix[0,1];
    Matrix[0,1]:=CosAngle*Matrix[0,1]-SinAngle*Matrix[0,2];
    Matrix[0,2]:=SinAngle*TempReal+CosAngle*Matrix[0,2];
    TempReal:=Matrix[1,1];
    Matrix[1,1]:=CosAngle*Matrix[1,1]-SinAngle*Matrix[1,2];
    Matrix[1,2]:=SinAngle*TempReal+CosAngle*Matrix[1,2];
    TempReal:=Matrix[2,1];
    Matrix[2,1]:=CosAngle*Matrix[2,1]-SinAngle*Matrix[2,2];
    Matrix[2,2]:=SinAngle*TempReal+CosAngle*Matrix[2,2]; End;

  If PgUpFlag Xor PgDnFlag Then Begin
    If PgUpFlag Then SinAngle:=SinPositiveAngle
    Else SinAngle:=-SinPositiveAngle;
    TempReal:=Matrix[0,0];
    Matrix[0,0]:=CosAngle*Matrix[0,0]-SinAngle*Matrix[0,1];
    Matrix[0,1]:=SinAngle*TempReal+CosAngle*Matrix[0,1];
    TempReal:=Matrix[1,0];
    Matrix[1,0]:=CosAngle*Matrix[1,0]-SinAngle*Matrix[1,1];
    Matrix[1,1]:=SinAngle*TempReal+CosAngle*Matrix[1,1];
    TempReal:=Matrix[2,0];
    Matrix[2,0]:=CosAngle*Matrix[2,0]-SinAngle*Matrix[2,1];
    Matrix[2,1]:=SinAngle*TempReal+CosAngle*Matrix[2,1]; End;



    If MinusFlag Then Distance:=Distance-DeltaDistance;
  If PlusFlag Then Distance:=Distance+DeltaDistance;
  If Distance>1.0 Then Distance:=1.0
  Else If Distance0.001 Then Distance:=0.001;

  If HomeFlag Then Begin Distance:=0.7;
    For I:=0 To 2 Do For J:=0 To 2 Do
 If I=J Then Matrix[I,J]:=32760.0 Else Matrix[I,J]:=0.0; End;

(* Keep looping till the user presses CTRL and BREAK, CTRL and C, or ESC. *)

Until ExitFlag;
```

```
ExitFlag:=false; (* reset flag so program does not end *)
     flagf1:=false;
     flagf2:=false;
     Flagf3:=false;
     MaximumBuffer:=1;

  (* clear screen again  *)
  For I:=MaximumBuffer Downto 0 Do Begin
     SetDrawingBufferVar.Buffer:=I;
     AcroMole.SetDrawingBuffer(SetDrawingBufferVar);

   DrawRectangleVar.ScreenX1:=MinimumScreenX;
   DrawRectangleVar.ScreenY1:=MinimumScreenY;
   DrawRectangleVar.ScreenX2:=MaximumScreenX;
   DrawRectangleVar.ScreenY2:=MaximumScreenY;
   DrawRectangleVar.color:=blue;AcroMole.DrawRectangle(DrawRectangleVar);
   end;
     (* clear screen again  *)
  For I:=MaximumBuffer Downto 0 Do Begin
     SetDrawingBufferVar.Buffer:=I;
     AcroMole.SetDrawingBuffer(SetDrawingBufferVar);

   DrawRectangleVar.ScreenX1:=MinimumScreenX;
   DrawRectangleVar.ScreenY1:=MinimumScreenY;
   DrawRectangleVar.ScreenX2:=MaximumScreenX;
   DrawRectangleVar.ScreenY2:=MaximumScreenY;
   DrawRectangleVar.color:=blue;AcroMole.DrawRectangle(DrawRectangleVar);
   end;

DrawOutlines; (* Setup screen for original look *)
end;{if f#flag}

  If MinusFlag Then Distance:=Distance-DeltaDistance;
  If PlusFlag Then Distance:=Distance+DeltaDistance;
  If Distance>1.0 Then Distance:=1.0
  Else If Distance0.001 Then Distance:=0.001;

  If HomeFlag Then Begin Distance:=0.7;
    For I:=0 To 2 Do For J:=0 To 2 Do
If I=J Then Matrix[I,J]:=32760.0 Else Matrix[I,J]:=0.0; End;


   If f12Flag then begin
     (* clear screen again  *)
  For I:=MaximumBuffer Downto 0 Do Begin
     SetDrawingBufferVar.Buffer:=I;
     AcroMole.SetDrawingBuffer(SetDrawingBufferVar);

   DrawRectangleVar.ScreenX1:=MinimumScreenX;
   DrawRectangleVar.ScreenY1:=MinimumScreenY;
   DrawRectangleVar.ScreenX2:=MaximumScreenX;
```

```
  DrawRectangleVar.ScreenY2:=MaximumScreenY;
  DrawRectangleVar.color:=blue;AcroMole.DrawRectangle(DrawRectangleVar);
  end;

     StopAcroMole;
     CloseGraph;


    Dispose(@WorldPointList);
    Dispose(@WorldEndPointList);
    Dispose(@WorldLineList);
    Dispose(@WorldPointListS1);
    Dispose(@WorldPointListS2);
    Dispose(@WorldPointListS3);


    worldpointlist:=nil;
    worldendpointlist:=nil;
    worldlinelist:=nil;
    worldpointlists1:=nil;
    worldpointlists2:=nil;
    worldpointlists3:=nil;

    Release(HeapOrg);
    Distance:=0.7;
    For I:=0 To 2 Do For J:=0 To 2 Do
If I=J Then Matrix[I,J]:=32760.0 Else Matrix[I,J]:=0.0;
   goto 888;  (* Restart Program *)
     end; {if f12Flag }

(* Keep looping till the user presses CTRL and BREAK, CTRL and C, or ESC. *)

Until ExitFlag; (* f#Flag test Repeat *)

(* Shut down AcroMole before returning to DOS. *)
Until ExitFlag; (* Main Program Repeat *)
StopAcroMole;
CloseGraph;
 Gd:=2;  Gm:=VGAHi;
   InitGraph(Gd,Gm,'');

closegraph;
End.
```

Appendix C

The first unit is MathStuff. This unit contains the following functions:

1. Ellipse x1x2 Top - Returns x2 on the bottom half of ellipse for a given x1.
2. Ellipse x1x2 Bottom - Returns x2 on the bottom half of ellipse for a given x1.
3. Ellipse x1x3 Top - Returns x3 on the bottom half of ellipse for a given x1.
4. Ellipse x1x3 Bottom - Returns x3 on the bottom half of ellipse for a given x1.
5. Ellipse x2x3 Top - Returns x2 on the bottom half of ellipse for a given x1.
6. Ellipse x2x3 Bottom - Returns x2 on the bottom half of ellipse for a given x1.

```
Unit MathStuf;
{$F+}
{$O+}
Interface
Uses Graph.Crt.Mathmat;

Procedure EllipseX1X2Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);   (* Creates Points for X1 vs X2 Ellipse*)
Procedure EllipseX1X2Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);  (* Creates Points for X1 vs X2 Ellipse*)

Procedure EllipseX1X3Top(var xx:real; sigmaStar:mathmat.matx;
      cvalue:mathmat.matx; xbarstar:mathmat.matx;
      var answer:real);   (* Creates Points for X1 vs X3 Ellipse*)
Procedure EllipseX1X3Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);   (* Creates Points for X1 vs X3 Ellipse*)

Procedure EllipseX2X3Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);   (* Creates Points for X2 vs X3 Ellipse*)
Procedure EllipseX2X3Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);   (* Creates Points for X2 vs X3 Ellipse*)


Procedure EllipseX2X1Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);   (* Creates Points for X2 vs X1 Ellipse*)
Procedure EllipseX2X1Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);  (* Creates Points for X2 vs X1 Ellipse*)

Procedure EllipseX3X1Top(var xx:real;sigmaStar:mathmat.matx;
      cvalue:mathmat.matx;xbarstar:mathmat.matx;
      var answer:real);   (* Creates Points for X3 vs X1 Ellipse*)
```

```
Procedure EllipseX3X1Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);  (* Creates Points for X3 vs X1 Ellipse*)

Procedure EllipseX3X2Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);  (* Creates Points for X3 vs X2 Ellipse*)
Procedure EllipseX3X2Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);  (* Creates Points for X3 vs X2 Ellipse*)


Implementation

Procedure EllipseX1X2Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var

  x2a,x2b,x2c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

x2a:=(sigmaStar.data[1,2]*sigmaStar.data[1,2])-sigmaStar.data[1,1]*sigmaStar.data[2,2];
x2b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[2,2]-2.0*(sigmaStar.data[1,2]*sigmaS-
tar.data[1,2]);
x2c:=(sigmaStar.data[1,2]*sigmaStar.data[1,2])-sigmaStar.data[1,1]*sigmaStar.data[2,2];
A1:=-sigmaStar.data[1,1]*xbarstar.data[2,1];
A2:=x2a*(xbarstar.data[1,1]*xbarstar.data[1,1]);
A3:=x2b*xx*xbarstar.data[1,1];
A4:=x2c*(xx*xx);
A5:=(cvalue.data[1,1]*cvalue.data[1,1])*(sigmaStar.data[1,1]*sigmaStar.data[1,1])*sigmaS-
tar.data[2,2];
A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[1,1]*sqr(sigmaStar.data[1,2]);
A7:=sigmaStar.data[1,2]*xbarstar.data[1,1]-sigmaStar.data[1,2]*xx;


  answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[1,1];
  End;

Procedure EllipseX1X2Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);


Var
  x2a,x2b,x2c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin
```

```
x2a:=sqr(sigmaStar.data[1,2])-sigmaStar.data[1,1]*sigmaStar.data[2,2];
x2b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[2,2]-(2.0*sqr(sigmaStar.data[1,2]));
x2c:=sqr(sigmaStar.data[1,2])-(sigmaStar.data[1,1]*sigmaStar.data[2,2]);
A1:=-sigmaStar.data[1,1]*xbarstar.data[2,1];
A2:=x2a*sqr(xbarstar.data[1,1]);
A3:=x2b*xx*xbarstar.data[1,1];
A4:=x2c*sqr(xx);
A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[1,1])*sigmaStar.data[2,2];
A6:=-1*sqr(cvalue.data[1,1])*sigmaStar.data[1,1]*sqr(sigmaStar.data[1,2]);
A7:=sigmaStar.data[1,2]*xbarstar.data[1,1]-(sigmaStar.data[1,2]*xx);

  answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[1,1];

  End;

Procedure EllipseX1X3Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin


 x3a:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 x3b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[3,3]-2.0*sqr(sigmaStar.data[1,3]);
 x3c:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 A1:=-sigmaStar.data[1,1]*xbarstar.data[3,1];
 A2:=x3a*sqr(xbarstar.data[1,1]);
 A3:=x3b*xx*xbarstar.data[1,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[1,1])*sigmaStar.data[3,3];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[1,1]*sqr(sigmaStar.data[1,3]);
 A7:=sigmaStar.data[1,3]*xbarstar.data[1,1]-sigmaStar.data[1,3]*xx;

answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[1,1];
  End;

Procedure EllipseX1X3Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);
Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin


 x3a:=sqr(sigmaStar.data[1,3])-(sigmaStar.data[1,1]*sigmaStar.data[3,3]);
 x3b:=(2.0*sigmaStar.data[1,1]*sigmaStar.data[3,3])-2.0*sqr(sigmaStar.data[1,3]);
 x3c:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
```

```
 A1:=-sigmaStar.data[1,1]*xbarstar.data[3,1];
 A2:=x3a*sqr(xbarstar.data[1,1]);
 A3:=x3b*xx*xbarstar.data[1,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[1,1])*sigmaStar.data[3,3];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[1,1]*sqr(sigmaStar.data[1,3]);
 A7:=sigmaStar.data[1,3]*xbarstar.data[1,1]-sigmaStar.data[1,3]*xx;

answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[1,1];

  End;


Procedure EllipseX2X3Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

 x3a:=sqr(sigmaStar.data[2,3])-sigmaStar.data[2,2]*sigmaStar.data[3,3];
 x3b:=2.0*sigmaStar.data[2,2]*sigmaStar.data[3,3]-2.0*sqr(sigmaStar.data[2,3]);
 x3c:=sqr(sigmaStar.data[2,3])-sigmaStar.data[2,2]*sigmaStar.data[3,3];
 A1:=-sigmaStar.data[2,2]*xbarstar.data[3,1];
 A2:=x3a*sqr(xbarstar.data[2,1]);
 A3:=x3b*xx*xbarstar.data[2,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[2,2])*sigmaStar.data[3,3];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[2,2]*sqr(sigmaStar.data[2,3]);
 A7:=sigmaStar.data[2,3]*xbarstar.data[2,1]-sigmaStar.data[2,3]*xx;

 answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[2,2];
  End;

Procedure EllipseX2X3Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin


 x3a:=sqr(sigmaStar.data[2,3])-sigmaStar.data[2,2]*sigmaStar.data[3,3];
 x3b:=2.0*sigmaStar.data[2,2]*sigmaStar.data[3,3]-2.0*sqr(sigmaStar.data[2,3]);
 x3c:=sqr(sigmaStar.data[2,3])-sigmaStar.data[2,2]*sigmaStar.data[3,3];
 A1:=-sigmaStar.data[2,2]*xbarstar.data[3,1];
 A2:=x3a*sqr(xbarstar.data[2,1]);
 A3:=x3b*xx*xbarstar.data[2,1];
```

```
  A4:=x3c*sqr(xx);
  A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[2,2])*sigmaStar.data[3,3];
  A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[2,2]*sqr(sigmaStar.data[2,3]);
  A7:=sigmaStar.data[2,3]*xbarstar.data[2,1]-sigmaStar.data[2,3]*xx;

answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[2,2];
  End;

Procedure EllipseX2X1Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x2a,x2b,x2c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

x2a:=(sigmaStar.data[1,2]*sigmaStar.data[1,2])-sigmaStar.data[2,2]*sigmaStar.data[1,1];
x2b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[2,2]-2.0*(sigmaStar.data[1,2]*sigmaS-
tar.data[1,2]);
x2c:=(sigmaStar.data[1,2]*sigmaStar.data[1,2])-sigmaStar.data[1,1]*sigmaStar.data[2,2];
A1:=-sigmaStar.data[2,2]*xbarstar.data[1,1];
A2:=x2a*(xbarstar.data[2,1]*xbarstar.data[2,1]);
A3:=x2b*xx*xbarstar.data[2,1];
A4:=x2c*(xx*xx);
A5:=(cvalue.data[1,1]*cvalue.data[1,1])*(sigmaStar.data[2,2]*sigmaStar.data[2,2])*sigmaS-
tar.data[1,1];
A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[2,2]*sqr(sigmaStar.data[1,2]);
A7:=sigmaStar.data[1,2]*xbarstar.data[2,1]-sigmaStar.data[1,2]*xx;


  answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[2,2];
  End;

Procedure EllipseX2X1Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x2a,x2b,x2c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

x2a:=sqr(sigmaStar.data[2,1])-sigmaStar.data[2,2]*sigmaStar.data[1,1];
x2b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[2,2]-2.0*(sigmaStar.data[2,1]*sigmaS-
tar.data[2,1]);
x2c:=(sigmaStar.data[1,2]*sigmaStar.data[1,2])-sigmaStar.data[1,1]*sigmaStar.data[2,2];
A1:=-sigmaStar.data[2,2]*xbarstar.data[1,1];
A2:=x2a*(xbarstar.data[2,1]*xbarstar.data[2,1]);
```

```
A3:=x2b*xx*xbarstar.data[2,1];
A4:=x2c*(xx*xx);
A5:=(cvalue.data[1,1]*cvalue.data[1,1])*(sigmaStar.data[2,2]*sigmaStar.data[2,2])*sigmaS-
tar.data[1,1];
A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[2,2]*sqr(sigmaStar.data[2,1]);
A7:=sigmaStar.data[2,1]*xbarstar.data[2,1]-sigmaStar.data[2,1]*xx;


  answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[2,2];


  End;

Procedure EllipseX3X1Top(var xx:real;sigmaStar:mathmat.matx;
      cvalue:mathmat.matx; xbarstar:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

 x3a:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 x3b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[3,3]-2.0*sqr(sigmaStar.data[1,3]);
 x3c:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 A1:=-sigmaStar.data[3,3]*xbarstar.data[1,1];
 A2:=x3a*sqr(xbarstar.data[3,1]);
 A3:=x3b*xx*xbarstar.data[3,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[3,3])*sigmaStar.data[1,1];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[3,3]*sqr(sigmaStar.data[1,3]);
 A7:=sigmaStar.data[1,3]*xbarstar.data[3,1]-sigmaStar.data[1,3]*xx;

answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[3,3];
  End;

Procedure EllipseX3X1Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);
Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

  x3a:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 x3b:=2.0*sigmaStar.data[1,1]*sigmaStar.data[3,3]-2.0*sqr(sigmaStar.data[1,3]);
 x3c:=sqr(sigmaStar.data[1,3])-sigmaStar.data[1,1]*sigmaStar.data[3,3];
 A1:=-sigmaStar.data[3,3]*xbarstar.data[1,1];
 A2:=x3a*sqr(xbarstar.data[3,1]);
 A3:=x3b*xx*xbarstar.data[3,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[3,3])*sigmaStar.data[1,1];
```

```
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[3,3]*sqr(sigmaStar.data[1,3]);
 A7:=sigmaStar.data[1,3]*xbarstar.data[3,1]-sigmaStar.data[1,3]*xx;


answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[3,3];

  End;


Procedure EllipseX3X2Top(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx;xbar-
star:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

 x3a:=sqr(sigmaStar.data[2,3])-sigmaStar.data[3,3]*sigmaStar.data[2,2];
 x3b:=2.0*sigmaStar.data[3,3]*sigmaStar.data[2,2]-2.0*sqr(sigmaStar.data[2,3]);
 x3c:=sqr(sigmaStar.data[2,3])-sigmaStar.data[3,3]*sigmaStar.data[2,2];
 A1:=-sigmaStar.data[3,3]*xbarstar.data[2,1];
 A2:=x3a*sqr(xbarstar.data[3,1]);
 A3:=x3b*xx*xbarstar.data[3,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[3,3])*sigmaStar.data[2,2];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[3,3]*sqr(sigmaStar.data[2,3]);
 A7:=sigmaStar.data[2,3]*xbarstar.data[3,1]-sigmaStar.data[2,3]*xx;

 answer:=-(A1+Sqrt(A2+A3+A4+A5+A6)+A7)/sigmaStar.data[3,3];
  End;

Procedure EllipseX3X2Bottom(var xx:real;sigmaStar:mathmat.matx;cvalue:mathmat.matx
      ;xbarstar:mathmat.matx;
      var answer:real);

Var
  x3a,x3b,x3c,A1,A2,A3,A4,A5,A6,A7:Real;

Begin

 x3a:=sqr(sigmaStar.data[2,3])-sigmaStar.data[3,3]*sigmaStar.data[2,2];
 x3b:=2.0*sigmaStar.data[3,3]*sigmaStar.data[2,2]-2.0*sqr(sigmaStar.data[2,3]);
 x3c:=sqr(sigmaStar.data[2,3])-sigmaStar.data[3,3]*sigmaStar.data[2,2];
 A1:=-sigmaStar.data[3,3]*xbarstar.data[2,1];
 A2:=x3a*sqr(xbarstar.data[3,1]);
 A3:=x3b*xx*xbarstar.data[3,1];
 A4:=x3c*sqr(xx);
 A5:=sqr(cvalue.data[1,1])*sqr(sigmaStar.data[3,3])*sigmaStar.data[2,2];
 A6:=-sqr(cvalue.data[1,1])*sigmaStar.data[3,3]*sqr(sigmaStar.data[2,3]);
 A7:=sigmaStar.data[2,3]*xbarstar.data[3,1]-sigmaStar.data[2,3]*xx;
```

```
answer:=(-A1+Sqrt(A2+A3+A4+A5+A6)-A7)/sigmaStar.data[3,3];
  End;


  End. {unit}
```

Appendix D

The second unit, MathMat contain several procedures for performing matrix algebra.

1. MatInvert - inverts a matrix.
2. MatMult - multiplies two matrices.
3. MatAdd - adds two matrices.
4. MatSub - subtracts two matrices.
5. MatMut_by_k - multiplies a matrix by a constant.
6. Zero_Matrix - creates a zero matrix.
7. Mat_Transpose - transposes a matrix.
8. Mat_Angment - combines two matrices or vectors into a single matrix.

```
unit MathMat;
{$F+}
{$O+}
interface
uses Crt,Dos,Graph;

 Const
  np=10;    {Matrix can be up to 10 x 10, modify np for larger matrics}

 Type
  RealArrayNPbyNP = ARRAY[1..np,1..np] of  real;
  matx = Record Data: RealArrayNPbyNP;
         Rows,Cols: Integer; end;

 Procedure MatInvert (var b,y: matx);   {y=invert of matrix b}

 Procedure MatMult (var a,b,c: matx);   {C=A*B}

 Procedure MatAdd (var a,b,c: matx);    {C=A+B}

 Procedure MatSub (var a,b,c: matx);    {C=A-B}

 Procedure Mat_k_Mult (var a,c: matx;   {C=k*A}
                  k: Real);  {k is a scalar}

 Procedure Mat_Zero (var a: matx);

 Procedure Mat_Transpose (var a,b: matx);  {b is transpose of a}

 Procedure MatWrite (var a: matx);      {Writes an rowxcol matrix}

 Procedure MatInput(var matrx: matx;
         cell_length: Integer;   {Width of each cell}
          dec_places: Integer;   {Number of Decimal Places per Cell}
              mat: String;    {Input Matrix Name}
              x,y: Integer);  {Location}
```

```
 Procedure MatAugment(var augmat,mat1,mat2: matx);

implementation

 Type
  RealArrayNP = ARRAY [1..np] of real;
  IntegerArrayNP = ARRAY [1..np] of integer;

var
 i,j: integer;

Procedure ludcmp (var a:     RealArrayNPbyNP;
                  n:    integer;
              var indx: IntegerArrayNP;
              var d:    real);

{Inversion Routine primarily based on routines from the book:
 Numerical Recipes in Pascal by William H. Press and others,
 Published by the Press Syndicate of the University of Cambridge,
 New York, 1989.
 Pages 42-46}

Const
 tiny = 1.0e-20;

Var
 k,j,imax,i:  integer;
 sum,dum,big:  real;
 vv: ^RealArrayNP;

Begin
 new(vv);
 d:=1.0;
 For i:=1 to n do begin
  big := 0.0;
  for j := 1 to n do
   if abs(a[i,j])>big then big :=abs(a[i,j]);
  if big = 0.0 then begin
   writeln ('pause in LUDCMP - singular matrix');
   readln
  end;
  vv^[i] := 1.0/big
 end;
 for j := 1 to n do begin
  for i := 1 to j-1 do begin
   sum :=a[i,j];
   for k := 1 to i-1 do
    sum := sum-a[i,k]*a[k,j];
   a[i,j] := sum
  end;
  big := 0.0;
```

```
    for i := j to n do begin
      sum := a[i,j];
      for k := 1 to j-1 do
        sum := sum-a[i,k]*a[k,j];
      a[i,j] := sum;
      dum := vv^[i]*abs(sum);
      if dum >= big then begin
        big := dum;
        imax :=i
      end
    end;
    if j <> imax then begin
      for k :=1 to n do begin
        dum := a[imax,k];
        a[imax,k] := a[j,k];
        a[j,k] := dum
      end;
      d := -d;
      vv^[imax] := dum
    end;
    indx[j] := imax;
    if a[j,j] = 0.0 then a[j,j] := tiny;
    if j<> n then begin
      dum := 1.0/a[j,j];
      for i := j+1 to n do
        a[i,j]:= a[i,j]*dum
    end
  end;
  dispose(vv)
end;

Procedure lubksb (var a: RealArrayNPbyNP;
                  n: integer;
             var indx: IntegerArrayNP;
               var b: RealArrayNP);

{Inversion Routine primarily based on routines from the book:
 Numerical Recipes in Pascal by William H. Press and others,
 Published by the Press Syndicate of the University of Cambridge,
 New York, 1989.
 Pages 42-46}

var
  j,ip,ii,i: integer;
  sum: real;

begin
  ii:=0;

  for i :=1 to n do begin
    ip := indx[i];
    sum := b[ip];
```

```
  b[ip] := b[i];
  if ii <> 0 then
    for j := ii to i-1 do
      sum := sum-a[i,j]*b[j]
  else if sum <> 0.0 then
    ii := i;
  b[i] := sum;
 end;
 for i := n downto 1 do begin
  sum := b[i];
  for j:= i+1 to n do
    sum := sum-a[i,j]*b[j];
  b[i] := sum/a[i,i];
 end;
end;

Procedure MatInvert (var b,y: matx);

{Inversion Routine primarily based on routines from the book:
 Numerical Recipes in Pasca' )y William H. Press and others,
 Published by the Press Syndicate of the University of Cambridge,
 New York, 1989.
 Pages 42-46}

var
 a:   RealArrayNPbyNP;
 i,j: integer;
 col: RealArrayNP;
 indx: IntegerArrayNP;
 d:   Real;

begin
 a := b.data;
 ludcmp(a,b.rows,indx,d);
 for j:= 1 to b.rows do begin
   for i:= 1 to b.rows do col[i] :=0.0;
   col[j] := 1.0;
   lubksb(a,b.rows,indx,col);
   for i := 1 to b.rows do y.data[i,j] := col[i]
 end;
 y.rows:=b.rows; y.cols:=b.cols;
end;

Procedure MatMult (var a,b,c: matx);     {C=A*B}

var
 row,col,i: Integer;
 sum: Real;

begin
 if a.cols  = b.rows then begin
   for col := 1 to b.cols do begin
```

```
    for row := 1 to a.rows do begin
     sum := 0;
     for i := 1 to b.rows do
      sum := sum + a.data[row,i]*b.data[i,col];
     c.data[row,col] := sum;
    end;
   end;
  c.rows := a.rows;
  c.cols := b.cols;
  end
  else begin
   writeln ('The number of columns of the first matrix must equal the');
   writeln ('number of rows of the second matrix in order to multiply');
   writeln ('matrices.')
  end;
end;


Procedure MatWrite (var a: matx);

var
 i,j: Integer;

begin
 writeln;
 for i := 1 to a.rows do begin
  for j := 1 to a.cols do
   write (a.data[i,j]:5,'  ');
  writeln (' ');
 end;
end;

Procedure MatAdd (var a,b,c: matx);    {C=A+B}

var
 i,j: Integer;

begin
 for i := 1 to a.rows do begin
  for j := 1 to a.cols do
   c.data[i,j] := a.data[i,j] + b.data[i,j];
 end;
 c.rows := a.rows;
 c.cols := a.cols;
end;


Procedure MatSub (var a,b,c: matx);    {C=A-B}

var
```

```
 i,j: Integer;

begin
 for i := 1 to a.rows do begin
  for j := 1 to a.cols do
   c.data[i,j] := a.data[i,j] - b.data[i,j];
 end;
 c.rows := a.rows;
 c.cols := a.cols;
end;


Procedure Mat_k_Mult (var a,c: matx;     {C=k*A}
                      k: Real);     {k is a scalar}

var
 i,j: Integer;

begin
 for i := 1 to a.rows do begin
  for j := 1 to a.cols do
   c.data[i,j] := k*a.data[i,j];
 end;
 c.rows := a.rows;
 c.cols := a.cols;
end;

Procedure Mat_Zero (var a: matx);

var
 m,n:  Integer;

begin
 for m := 1 to a.rows do
  for n := 1 to a.cols do
   a.data[m,n] := 0;
end;

Procedure Mat_Transpose (var a,b: matx);

var
 i,j:  Integer;

begin
 for i := 1 to a.rows do begin
  for j := 1 to a.cols do
   b.data[j,i] := a.data[i,j];
  end;  {for i begin}
 b.rows := a.cols;
 b.cols := a.rows;
end; {Mat_Transpose}
```

```
Procedure MatInput(var matrx: matx;
          cell_length: Integer;
           dec_places: Integer;
               mat: String;    {Input Matrix Name}
               x,y: Integer);  {Location}

procedure cursor_to_cell(i,j: integer);   {Move Cursor to Current Cell}

begin
 case j of
  0   : GotoXY(length(mat)+3,i+1);
  1   : GotoXY(length(mat)+5,i+1);
 else   GotoXY(length(mat)+j*cell_length,i+1)
 end; {case j}
end; {cursor_to_cell}


procedure brackets;

const
 open_top_bracket = chr(218);         {ASCII Codes}
 open_bottom_bracket = chr(192);
 vertical_bar = chr(179);
 close_top_bracket = chr(191);
 close_bottom_bracket = chr(217);

var
 i,j: integer;

begin
 GotoXY(1,trunc((matrx.rows+3)/2));
 Write (mat,'=');
 for i := 0 to matrx.rows+1 do begin
  j := 0;
  cursor_to_cell(i,j);
  If i = 0 then write(open_top_bracket)
  else If i = matrx.rows+1 then write(open_bottom_bracket)
     else write(vertical_bar);

  j := matrx.cols+1;
  cursor_to_cell(i,j);
  If i = 0 then write(close_top_bracket)
  else If i = matrx.rows+1 then write(close_bottom_bracket)
     else write(vertical_bar);

 end;  {loop i}
 GotoXY(2,matrx.rows +4);
 Write('Press F10 when finished');
end;  {brackets}
```

```
procedure SetUp(x,y:integer);  {Location of window}

var
  width: integer;

begin
  If (y+matrx.rows +2)>24 then begin
    writeln ('Sorry, cursor position is to near the bottom of the screen.');
    writeln ('x=',x,'  y=',y);
    writeln ('Hit  to Continue');
    readln;
    end
  else
   If (x+cell_length*(matrx.cols+1)+3+length(mat))>80 then begin
     writeln ('Sorry, cursor position is to near the right edge of the screen.');
     writeln ('x=',x,'  y=',y);
     writeln ('Hit  to Continue');
     readln;
     end;
  Width := cell_length*(matrx.cols+1)+length(mat)+3;
  If Width < 28 then Width := 28;
  Window(x,y,x+Width,y+matrx.rows +3);
  TextBackground(Blue);
  ClrScr;
  brackets;
end; {Setup}


procedure cell_text;        {Set Colors to Highlight Current Cell}

begin
  TextBackground(LightGray);
  TextColor(Black);
end; {cell_text}


procedure normal_text;      {Reset Colors}

begin
  TextBackground(Blue);
  TextColor(LightGray);
end; {normal_text}


procedure input_number(var matrx: matx);  {input matrix}

const
  left_arrow = 75;    {0 is returned first for arrow keys!}
  right_arrow = 77;
```

```
 up_arrow = 72;
 down_arrow = 80;
 backspace = 8;
 return = 13;
 escape = 27;
 plus = 43;
 minus = 45;
 F10 = 68;
 decimal = 46;

var
 key,key2 : char;
 k,dec,exit_cell,exit_procedure,errorcode  : Integer;
 number : string;     {input string}


procedure cell_write(number:string);

var
 k,errorcode: integer;
 s: string;
 x: real;

begin
  val(number,x,errorcode);
  str(x:cell_length:dec_places,s);
  write(s);
end; {cell_write}


procedure exec_return (var number_string: string;
                  var value: Real;
                  var exit_cell: integer);
var
 k:  integer;
begin
  val(number,value,errorcode);
  exit_cell := 1;       {input is complete}
  If errorcode <> 0 then writeln ('error in exec_return procedure');
  Cursor_to_Cell(i,j);
  normal_text;
  cell_write(number);
end;  {exec_return}


procedure too_long;

var
 h,m,s,s100,s_delay,s_count: Word;

begin
```

```
  GotoXY(2,matrx.rows +3);
  Write('Input Limited to ',Cell_Length,' digits');
  GetTime(h,m,s,s100);
  s_delay := s+3;
  GotoXY(2,matrx.rows +3);
  repeat
   GetTime(h,m,s,s100);
  until s >= s_delay;
  TextBackground(Blue);
  Write('                    ');
end; {too_long}

procedure char_ok(key:char;
         var number: string);
var
 x,y,i: integer;

begin
 write(key);
 number := number + key;
 If length(number) = 1 then begin
  x := WhereX;
  y := WhereY;
  For i := 2 to Cell_Length do write(' ');
  GotoXY(x,y);
 end; {if}
end; {char_ok}

function strg(x:real) : string;

var
 s: string;

begin
 str(x:cell_length:dec_places,s);
 strg := s;
end; {strg}

procedure WrapAround;

begin
 if i > matrx.rows then i := 1;
 if i < 1 then i := matrx.rows ;
 if j > matrx.cols then j := 1;
 if j < 1 then j := matrx.cols ;
end; {WrapAround}

procedure ReWrit(i,j: Integer);

begin
  Cursor_to_Cell(i,j);
  normal_text;
```

```
  If length(number) = 0 then cell_write(strg(matrx.data[i,j]))
  else cell_write(number);
end; {ReWrit}


procedure initialize;

var
 i,j: integer;
 number: string;

begin
 number := '';
 for i := 1 to matrx.rows  do begin
  for j := 1 to matrx.cols  do ReWrit(i,j);
  end; {for i begin}
end; {Initialize}

(***************************************)
(****  Main Procedure Input_Number ****)
(***************************************)

begin
 Initialize;
 i := 1;
 j := 1;
 exit_procedure := 0;
 repeat
  number := '';
  exit_cell := 0;
  dec := 0;
  cursor_to_cell(i,j);
  cell_text;
  cell_write(strg(matrx.data[i,j]));
  cursor_to_cell(i,j);
  repeat
   key := readkey;
   if Ord(key) = 0 then key2 := readkey;     {Read Extended Key Code}
   case Ord(key) of
    48..57     : If length(number) < Cell_Length then char_ok(key,number)
   { Numbers }     else too_long;

    return     : If length(number) > 0 then begin
                exec_return(number,matrx.data[i,j],exit_cell);
                If i = matrx.rows  then begin
                 i := 1;            {was on bottom row}
                 j := j+1;
                 end  {if i begin}
                else i := i+1;
                end {if length begin}
               else begin
                ReWrit(i,j);
```

```
        i := i+1;
        exit_cell := 1;
        end; {else begin}

decimal   : If dec = 0 then
        If length(number) < Cell_Length then begin
         char_ok(key,number);
         dec := 1;
         end {begin}
        else too_long;

  0     : begin
         case Ord(key2) of
          left_arrow  : begin
                  If length(number)>0 then
                  exec_return(number,matrx.data[i,j],exit_cell)
                  else ReWrit(i,j);
                  j := j-1;
                 end;
          right_arrow : begin
                  If length(number)>0 then
                  exec_return(number,matrx.data[i,j],exit_cell)
                  else ReWrit(i,j);
                  j := j+1;
                 end;
          up_arrow    : begin
                  If length(number)>0 then
                   exec_return(number,matrx.data[i,j],exit_cell)
                  else ReWrit(i,j);
                  i := i-1;
                 end;
          down_arrow  : begin
                  If length(number)>0 then
                   exec_return(number,matrx.data[i,j],exit_cell)
                  else ReWrit(i,j);
                  i := i+1;
                 end;
          F10         : begin
                  If length(number) > 0 then
                   exec_return(number,matrx.data[i,j],exit_cell)
                  else ReWrit(i,j);
                  exit_procedure := 1;
                  GotoXY(2,matrx.rows +4);
                  Write('                  ');
                 end;
         end;  {case Ord(key2)}
         exit_cell := 1;
        end;  {begin}
backspace  : begin
        number := Copy (number,1,(length(number)-1));
        GotoXY(WhereX-1,WhereY);
        write (' ');
```

```
              GotoXY(WhereX-1,WhereY);
            end;

    escape    : begin
              number := '';
              Cursor_to_Cell(i,j);
              for k := 1 to Cell_Length do write (' ');
              Cursor_to_Cell(i,j);
            end;

    plus      : If length(number) = 0 then char_ok(key,number);

    minus      : If length(number) = 0 then char_ok(key,number);


    else      ;   {Do Nothing if other keys are pressed}

   end; {case Ord(key)}
   WrapAround;
  until exit_cell = 1;
  normal_text;
 until exit_procedure =1;
end; {input_number}


begin
 Setup(x,y);
 input_number(matrx);
 Window(1,1,80,25);
end; {M_Input}

Procedure MatAugment(var augmat,mat1,mat2: matx);

var
 i,j: Integer;

begin
 for i := 1 to mat1.rows do begin
  for j := 1 to (mat1.cols+mat2.cols) do
  if j = mat1.cols then augmat.data[i,j] := mat1.data[i,j]
  else augmat.data[i,j] := mat2.data[i,j-mat1.cols];
 end; {for i}
 augmat.rows := mat1.rows;
 augmat.cols := mat1.cols+mat2.cols;
end; {MatAugment}

end. {unit}
```

Appendix E

The third, unit RGraphma uses procedures developed by Steve Pearce (cite thesis).

1. GMatInput - displays a matrix and gives the user ability the to change its values.
2. GMatWrite - displays a matrix without users ability to change its values.

```
unit RGraphMa;
{$F+}
{$O+}
interface

uses Crt,Dos,Graph,MathMat;

 Procedure GMatInput(var matrx: matx;
            cell_length: Integer;
             dec_places: Integer;
                  mat: String;    {Input Matrix Name}
                  x,y: Integer;   {Location}
              MatColor: Word;      {Color of Matrix}
         MatBackground: Word;      {Color of Matrix Background}
         MatForeground: Word);     {Color of Characters}

 Procedure GMatWrite(var matrx: matx;
            cell_length: Integer;
             dec_places: Integer;
                  mat: String;    {Input Matrix Name}
                  x,y: Integer;   {Location}
              MatColor: Word;      {Color of Matrix}
         MatBackground: Word;      {Color of Matrix Background}
         MatForeground: Word);     {Color of Characters}

implementation
type
  Intarray=array[1..6,1..6] of integer;
var
 i,j: integer;
 ViewPort: ViewPortType;


(*********************************************************************)
(*** The following routines are used in GMatInput and GMatWrite ***)
(*********************************************************************)

procedure cursor_to_cell(i,j: integer;   {Move Cursor to Current Cell}
                x,y: integer;   {Location of Entire Matrix}
                W,H: integer;   {Width & Height of Char Set}
```

```
                    mat: string;     {Name of Matrix}
                    cell_length:integer);  {# of digits per cell}

begin
 case j of
  0  : SetViewPort(X+(W-4)*length(mat),Y+H*i,
             X+W*(cell_length+length(mat)),Y+H*(1+i)-1,False);
  1  : SetViewPort(Round(X+(W-4)*Length(mat)+15),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+cell_length*W),Y+H*(1+i)-1,False);
  2  : SetViewPort(Round(X+(W-4)*Length(mat)+20+50),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+60+cell_length*W),
             Y+H*(1+i)-1,False);
  3  : SetViewPort(Round(X+(W-4)*Length(mat)+25+100),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+110+cell_length*W),
             Y+H*(1+i)-1,False);

  4  :SetViewPort(Round(X+(W-4)*Length(mat)+25+150),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+160+cell_length*W),
             Y+H*(1+i)-1,False);
  5  :SetViewPort(Round(X+(W-4)*Length(mat)+25+200),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+210+cell_length*W),
             Y+H*(1+i)-1,False);
  6  :SetViewPort(Round(X+(W-4)*Length(mat)+25+250),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+260+cell_length*W),
             Y+H*(1+i)-1,False);
  7  :SetViewPort(Round(X+(W-4)*Length(mat)+25+300),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+310+cell_length*W),
             Y+H*(1+i)-1,False);
  8  :SetViewPort(Round(X+(W-4)*Length(mat)+25+350),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+360+cell_length*W),
             Y+H*(1+i)-1,False);
  9  :SetViewPort(Round(X+(W-4)*Length(mat)+25+400),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+410+cell_length*W),
             Y+H*(1+i)-1,False);
  10  :SetViewPort(Round(X+(W-4)*Length(mat)+25+450),Y+H*i,
             Round(X+(W-4)*Length(mat)+25+460+cell_length*W),
             Y+H*(1+i)-1,False);


 else  Writeln(output,'Look at else in Cursor_Cell in GraphMat');

 end; {case j}

 GetViewSettings(ViewPort);


end; {cursor_to_cell}


procedure brackets(x,y: integer;    {Location of Entire Matrix}
           W,H: integer;   {Width & Height of Char Set}
           mat: string;    {Name of Matrix}
```

```
            matrx: matx;       {Matrix containing data}
        cell_length: integer;   {# of digits per cell}
              F10: Boolean;   {Press F10 or Not}
       BracketColor: Word;     {Color of Bracket}
      MatBackground: Word);      {Color of Background}
var
 i,j: integer;
 TLX,TLY,BLX,BLY,TRX,TRY,BRX,BRY: integer;
 UpperTitle,LowerTitle,mat2: string;
 Upper: Boolean;
 OldColor: Word;
 OldStyle: TextSettingsType;

begin
 mat2:='';
 GetTextSettings(OldStyle);     {So we can set them back}
 OldColor:=GetColor;
 SetTextJustify(1,1);          {Center Horizontally & Vertically}
 SetColor(15);

  {The routine below looks for a decimal point and assumes that characters
  following the decimal point will be placed as a subscript.
  Only 1 level of subscripting is supported.}

 Upper:=True;  {Title starts in normal Text}
 UpperTitle:=''; LowerTitle:='';

 for i:=1 to Length(mat) do
 If Copy(mat,i,1) = '.' then Upper:=Not Upper
   else   Case Upper of
          True  : begin UpperTitle:=UpperTitle+Copy(mat,i,1);
                  LowerTitle:=LowerTitle+' '; end;
          False : begin LowerTitle:=LowerTitle+Copy(mat,i,1);
                  UpperTitle:=UpperTitle+' '; end;
 end;

 If length(mat)>0 then begin mat2:=UpperTitle+'='; LowerTitle:=LowerTitle+' '; end;
 TLX := Round(X+(W-4)*(Length(mat)+2)); TLY := Y;
 TRX := TLX+W*(matrx.cols*Cell_Length+4+ord(matrx.cols>2)*1); TRY:=Y;
 BLX := TLX; BLY:= Y+round(H*(matrx.rows+1.5));
 BRX := TRX; BRY:= BLY;
 OutTextXY (x,y+Round(0.5*H*(matrx.rows+1.5)),mat2);
 OutTextXY (x,y+Round(0.5*H*(matrx.rows+1.5))+4,LowerTitle);
 SetLineStyle(0,0,3); {Solid, No Pattern, Thick}
 SetColor(BracketColor);
 MoveTo(TLX+W,TLY); LineTo(TLX,TLY); LineTo(BLX,BLY); LineTo(BLX+W,BLY);
 MoveTo(TRX-W,TRY); LineTo(TRX,TRY); LineTo(BRX,BRY); LineTo(BRX-W,BRY);
 SetColor(15);
 If F10 then begin
  SetFillStyle(1,MatBackground);
  Bar((BLX+BRX) div 2-W*7, Round(BLY+2.5*H),(BLX+BRX) div 2+W*7,
Round(BLY+3.5*H));
```

```
   OutTextXY ((BLX+BRX) div 2, BLY+3*H, 'F10 to finish'); end
  else begin
   SetFillStyle(1,MatBackground);
   Bar(TLX+2,TLY+2,BRX-2,BRY-2); end;



 SetTextJustify(OldStyle.Horiz, OldStyle.Vert);
 SetColor(OldColor);
end; {brackets}


procedure SetUp(x,y:integer;
          W,H: integer;   {Width & Height of Char Set}
          mat: string;    {Name of Matrix}
        matrx: matx;      {Matrix containing data}
  cell_length:integer;    {# of digits per cell}
          F10: Boolean);  {Press F10 or Not}


begin
 SetViewPort(0,0,GetMaxX,GetMaxY,True);
 If (y+(2+matrx.rows*H))>GetMaxY then begin
   Writeln(output,'Sorry, cursor position is to near the bottom of the screen.');
    Writeln(output, 'X=',X,' (W-4)=',W-4,' length(mat)=',length(mat),' cell_length=',
    cell_length,' matrx.cols=',matrx.cols);
   Writeln(output,'Hit  to Continue');
   readln;
   closegraph; Halt;
   end
 else
  If Round(X+(W-4)*(Length(mat)+2))+W*(matrx.cols*Cell_Length+2+
    ord(matrx.cols>2)*2)>GetMaxX then begin
    Writeln(output, 'Sorry, cursor position is to near the right edge of the screen.');
    Writeln(output, 'X=',X,' (W-4)=',W-4,' length(mat)=',length(mat),' cell_length=',
    cell_length,' matrx.cols=',matrx.cols);
    Writeln(output, 'Hit  to Continue');
    readln;
    closegraph; halt;
    end;
end; {Setup}

procedure cell_write(number:string;
          dec_places:integer;
         cell_length:integer);

var
 k,errorcode,dplaces: integer;
 s: string;
 x: real;
 OldStyle: TextSettingsType;
```

```
begin
  GetTextSettings(OldStyle);     {So we can set them back}
  SetTextJustify(0,2);           {Center Horizontally & Vertically}
  dplaces:=dec_places;
  val(number,x,errorcode);
  repeat
    str(x:cell_length:dplaces,s);
    if length(s)>cell_length then Dec(dplaces);
  until ((length(s)=cell_length) or (dplaces
  OutTextXY(0,1,s);
  SetTextJustify(OldStyle.Horiz, OldStyle.Vert);

end; {cell_write}


function strg(x:real;
    cell_length:integer;
     dec_places:integer) : string;

var
  s: string;

begin
  str(x:cell_length:dec_places,s);
  strg := s;
end; {strg}


Procedure GMatInput(var matrx: matx;
              cell_length: Integer;
               dec_places: Integer;
                     mat: String;    {Input Matrix Name}
                     x,y: Integer;   {Location}
                 MatColor: Word;     {Color of Matrix}
          MatBackground: Word;       {Color of Matrix Background}
          MatForeground: Word);      {Color of Characters}

var
  H: Integer;   {Height in Pixels of 1 character}
  W: Integer;   {Width in Pixels of 1 character}
 CM: Integer;   {Center of Matrix (X-wise)}
  ViewPort2: ViewPortType;
  OldStyle2: TextSettingsType;


procedure Cursor_Cell(i,j: integer);

begin
  cursor_to_cell(i,j,x,y,W,H,mat,cell_length);
end; {procedure Cursor_Cell}
```

```
procedure Cell_Rite(number:string);
begin
 Cell_Write(number,dec_places,cell_length);
end; {Procedure Cell_Rite}

procedure cell_text;        {Set Colors to Highlight Current Cell}

begin

 With ViewPort do begin
  SetFillStyle(1,       {Solid Pattern}
         7);      {Light Gray is Highlighted Cell Background}
  Bar(0,0,X2-X1,Y2-Y1); end;
 SetColor(15); {White is the Highlighted Cells' Foreground}
end; {cell_text}


procedure normal_text(Cell_Color:Word);       {Reset Colors}

begin
 SetColor(Cell_Color); {Normal Cell Foreground}
 With ViewPort do begin
  SetFillStyle(1, {Solid Pattern}
         MatBackground); {Blue is the cell background}
  Bar(0,0,X2-X1,Y2-Y1); end;
end; {normal_text}



procedure input_number(var matrx: matx);  {input matrix}

const
 left_arrow = 75;    {0 is returned first for arrow keys!}
 right_arrow = 77;
 up_arrow = 72;
 down_arrow = 80;
 backspace = 8;
 return = 13;
 escape = 27;
 plus = 43;
 minus = 45;
 F10 = 68;
 decimal = 46;

var
 key,key2 : char;
 k,decimals,exit_cell,exit_procedure,errorcode  : Integer;
 number : string;    {input string}
 TLX,TRX,BLY: Integer;
```

```
procedure exec_return (var number_string: string;
                var value: Real;
                var exit_cell: integer);
var
 k: integer;
begin
 val(number,value,errorcode);
 exit_cell := 1;      {input is complete}
 If errorcode <> 0 then OutText ('error in exec_return procedure');
 Cursor_Cell(i,j);
 normal_text(7);
 Cell_Rite(number);
end; {exec_return}


procedure too_long;

var
 ViewOld: ViewPortType;
 By,Rx,Lx: Integer;
 msg,cs: String;
 OldStyle: TextSettingsType;

begin
 GetTextSettings(OldStyle);     {So we can set them back}
 GetViewSettings(ViewOld);
 SetViewPort(0,0,GetMaxX,GetMaxY,True);
 By:= Y+round(H*(matrx.rows+1.5));
 Lx := Round(X+(W-4)*(Length(mat)+2));
 Rx := Round(Lx+W*(matrx.cols*Cell_Length+5));

 SetFillStyle(1,        {Solid Pattern}
        MatBackground);
 Bar(Round((Lx+Rx)/2-
12*W),Trunc(By+3.5*H),Round((Lx+Rx)/2+12*W),Round(By+2.5*H));
 SetTextJustify(1,1);
 Str(Cell_Length,cs);
 msg:='to '+cs+' digits';

 OutTextXY ((Lx+Rx) div 2,By+H,'Input Limited');
 OutTextXY ((Lx+Rx) div 2,By+2*H,msg);
 OutTextXY ((Lx+Rx) div 2, By+3*H,'(Press RETURN)');
 readln;
 SetFillStyle(1,        {Solid Pattern}
        MatBackground);
 Bar(Round((Lx+Rx)/2-7*W),Trunc(By+0.5*H),Round((Lx+Rx)/2+7*W),Round(By+3.5*H));
 OutTextXY ((LX+RX) div 2, BY+3*H, 'F10 to finish');
 SetTextJustify(OldStyle.Horiz, OldStyle.Vert);

 With ViewOld do SetViewPort(X1,Y1,X2,Y2,False);
end; {too_long}
```

```
procedure char_ok(key:char;
          var number: string);
var
 x,y,i: integer;
 OldStyle: TextSettingsType;

begin
 GetTextSettings(OldStyle);     {So we can set them back}
 SetTextJustify(0,2);           {Center Horizontally & Vertically}

 number := number + key;
 If length(number) = 1 then     {Blank out old number}
 With ViewPort do begin
  SetFillStyle(1,               {Solid Pattern}
         MatBackground); {Highlighted Cell Background}
  Bar(0,0,X2-X1,Y2-Y1); end;

 OutTextXY(GetX,1,key); MoveRel(W,0);
 With OldStyle do       {Reset TextSettings}
  begin
   SetTextJustify(Horiz, Vert);
   SetTextStyle(Font, Direction, CharSize);
  end;

end; {char_ok}

function strg(x:real) : string;

var
 s: string;

begin
 str(x:cell_length:dec_places,s);
 strg := s;
end; {strg}

procedure WrapAround;

begin
 if i > matrx.rows  then i := 1;
 if i < 1 then i := matrx.rows ;
 if j > matrx.cols  then j := 1;
 if j < 1 then j := matrx.cols ;
end; {WrapAround}

procedure ReWrit(i,j: Integer;
         number: string);

begin
  Cursor_Cell(i,j);
  normal_text(7);
  If length(number) = 0 then Cell_Rite(strg(matrx.data[i,j]))
```

```
  else Cell_Rite(number);
end; {ReWrit}


procedure initialize;

var
 i,j: integer;
 number: string;

begin
 number := '';
 for i := 1 to matrx.rows  do begin
   for j := 1 to matrx.cols  do rewrit(i,j,number);
   end; {for i begin}
end; {Initialize}

(****************************************)
(****   Main Procedure Input_Number ****)
(****************************************)

begin
 Assign(output,'prn');
 Rewrite(output);
 Initialize;
 i := 1;
 j := 1;
 exit_procedure := 0;
 repeat
  number := '';
  exit_cell := 0;
  decimals := 0;
  Cursor_Cell(i,j);
  cell_text;
  Cell_Rite(strg(matrx.data[i,j]));
  matrx.data[i,j]:=matrx.data[j,i];

  Cursor_Cell(i,j);

  repeat
   key := readkey;
   if Ord(key) = 0 then key2 := readkey;     {Read Extended Key Code}
 {   Writeln(output,'key 1=',key,'  key2=',key2);  }
   case Ord(key) of
    48..57     : If length(number) < Cell_Length then char_ok(key,number)
   { Numbers }      else too_long;

    return      : If length(number) > 0 then begin
                 exec_return(number,matrx.data[i,j],exit_cell);
                 If i = matrx.rows  then begin
                  i := 1;            {was on bottom row}
                  j := j+1;
```

```
          end  {if i begin}
         else i := i+1;
         end {if length begin}
        else begin
         rewrit(i,j,number);

         i := i+1;
         exit_cell := 1;
         end; {else begin}

decimal   : If decimals = 0 then
         If length(number) < Cell_Length then begin
          char_ok(key,number);
          decimals := 1;
          end {begin}
         else too_long;

  0     : begin
          case Ord(key2) of
           left_arrow  : begin
                     If length(number)>0 then
                     exec_return(number,matrx.data[i,j],exit_cell)
                     else rewrit(i,j,number);
                     j := j-1;
                    end;
           right_arrow : begin
                     If length(number)>0 then
                     exec_return(number,matrx.data[i,j],exit_cell)
                     else rewrit(i,j,number);
                     j := j+1;
                    end;
           up_arrow    : begin
                     If length(number)>0 then
                      exec_return(number,matrx.data[i,j],exit_cell)
                     else rewrit(i,j,number);
                     i := i-1;
                    end;
           down_arrow  : begin
                     If length(number)>0 then
                      exec_return(number,matrx.data[i,j],exit_cell)
                     else rewrit(i,j,number);
                     i := i+1;
                    end;
           F10       : begin
                     If length(number) > 0 then
                      exec_return(number,matrx.data[i,j],exit_cell)
                     else rewrit(i,j,number);
                     exit_procedure := 1;
                    end;
          end;  {case Ord(key2)}
          exit_cell := 1;
         end;  {begin}
```

```
    backspace  : If length(number)>0 then begin
               If number[length(number)]='.' then decimals:=0;
               number := Copy (number,1,(length(number)-1));
               SetFillStyle(1,        {Solid Pattern}
                7);       {Light Gray is Highlighted Cell Background}
               Bar(GetX-W,GetY+H-1,GetX,GetY);
               MoveRel(-W,0);
              end;

    escape     : begin
               number := '';
               With ViewPort do begin
                 SetFillStyle(1,   {Solid Pattern}
                        7);  {Light Gray is Highlighted Cell Background}
                 Bar(0,0,X2-X1,Y2-Y1); end;
               Cursor_Cell(i,j);
              end;

    plus      : If length(number) = 0 then char_ok(key,number);

    minus      : If length(number) = 0 then char_ok(key,number);


    else      ;   {Do Nothing if other keys are pressed}

   end; {case Ord(key)}
   WrapAround;
  until exit_cell = 1;
 until exit_procedure =1;
 SetViewPort(0,0,GetMaxX,GetMaxY,True);
 TLX := X-2*W; TRX := TLX+W*24;
 BLY:= Y+round(H*(matrx.rows+1.5));
 SetFillStyle(1,1); {Solid, Blue}
 Bar(Round((TLx+TRx)/2-
15*W),Trunc(BLy+3.5*H),Round((TLx+TRx)/2+10*W),Round(BLy+2.5*H));
 close(output);
end; {input_number}


begin
 H:= TextHeight('H');
 W:= TextWidth('W');
 CM:= X+(W*(Length(mat)+3)+W*(matrx.cols*Cell_Length+1) div 2);
 GetTextSettings(OldStyle2);
 GetViewSettings(ViewPort2);

 SetUp(x,y,W,H,mat,matrx,cell_length,True);
 brackets(x,y,W,H,mat,matrx,cell_length,True,MatColor,MatBackground);
 input_number(matrx);
 With ViewPort2 do SetViewPort(X1,Y1,X2,Y2,Clip);
 With OldStyle2 do
  begin
```

```
    SetTextJustify(Horiz, Vert);
    SetTextStyle(Font, Direction, CharSize);
   end;


end; {M_Input}


Procedure GMatWrite(var matrx: matx;
             cell_length: Integer;
              dec_places: Integer;
                   mat: String;    {Input Matrix Name}
                   x,y: Integer;   {Location}
               MatColor: Word;       {Color of Matrix}
           MatBackground: Word;      {Color of Matrix Background}
           MatForeground: Word);     {Color of Characters}

var
 H: Integer;    {Height in Pixels of 1 character}
 W: Integer;    {Width in Pixels of 1 character}
 ViewPort: ViewPortType;
 OldStyle: TextSettingsType;


begin
 GetTextSettings(OldStyle);
 GetViewSettings(ViewPort);
 W:=TextWidth('W');
 H:=TextHeight('H');
 SetUp(x,y,W,H,mat,matrx,cell_length,False);
 brackets(x,y,W,H,mat,matrx,cell_length,False,MatColor,MatBackground);
 SetColor(MatForeGround);
    for i:=1 to matrx.rows do for j:=1 to matrx.cols do begin
  cursor_to_cell(i,j,x,y,W,H,mat,cell_length);
  Cell_Write (strg(matrx.data[i,j],cell_length,dec_places),
          dec_places,cell_length);
  end;
 With ViewPort do SetViewPort(X1,Y1,X2,Y2,Clip);
 With OldStyle do
  begin
   SetTextJustify(Horiz, Vert);
   SetTextStyle(Font, Direction, CharSize);
  end;


end; {procedure GMatWrite}

end. {unit}
```

Appendix F

The final unit, Get_Data contains the program's main input sections and performs all calculations other than the points which make up the actual ellipses.

This unit contains the following procedures:

1. Jac - which is a version of the Jacobi algorithm for calculating eigenvalues and eigenvectors of symmetric matrices.
2. Get_Data keyboard - receives the empirical sample data for the x matrix directly from the keyboard and then performs all necessary calculations.
3. GenerateData - generates theoretical populations for a given μ and Σ.
4. ReadDataEmpirical - reads empirical sample data from disk.
5. ReadTestData - reads empirical sample data from disk and ??? data from keyboard, and then performs a test of the hypothesis $H_0 : \mu_0 = \bar{x}$.

```
 Unit Get_Data;
 {$F+}
 {$O+}

Interface


Uses mathmat,RGraphma,graph,crt;

Const
Mp=4;
nmat=1;

  Increment=0.05;                {Angle Change}
Type
  IntegerArray3 = Array [1..3] of integer;
  RealArrayMpbyMp=Array[1..MP,1..MP] of Real;
  RealArrayMp=Array[1..Mp] OF Real;
  Real3by12array=array[1..3,1..12] of real;
 Procedure GetDataKeyboard(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
       var cvalue,Mu,xbar:mathmat.matx;
              var look:boolean;Var EigenValues,eigenvectors,SbyN,
              evalstar,evecstar:mathmat.matx;var ordp:integer;var rmatrix:mathmat.matx);


 Procedure GenerateData(var txbar,xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
       var cvalue,Mu,xbar:mathmat.matx;
              var look:boolean;Var EigenValues,eigenvectors,SigmabyN,
              evalstar,evecstar:mathmat.matx;var ordp:integer;
              var sigma,sbyn,rmatrix:mathmat.matx);

 Procedure ReadDataEmpirical(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
       var cvalue,Mu,xbar:mathmat.matx;
              var look:boolean;Var EigenValues,eigenvectors,SbyN,
```

```
            evalstar,evecstar:mathmat.matx; Var ordp:integer;var rmatrix:mathmat.matx);

Procedure ReadDataTest(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
      var cvalue,Mu,xbar:mathmat.matx;
            var look:boolean;Var.EigenValues,eigenvectors,SbyN,
            evalstar,evecstar:mathmat.matx;var ordp:integer;var rmatrix:mathmat.matx);


Implementation
Var
i,j,k,kk,l,ll,nrot,ch:integer;
a,b,c,v,ad:RealArrayMpbyMp;
d,r:RealArrayMp;

xdata_trans,identity,one,one_trans,onebyoneT,temp,
      temp2,temp3,DMatrix,Dinverse,RMatrix,Z,
      Eigenvalues,eigenvectors,Cov,DsqrtEigenVals,N,Q:mathmat.matx;
grDriver,grMode,ErrCode,gd,gm:integer;
oneovern,oneovernminusone:real;
choice:string;
ordchoice,code:integer;

Procedure Jac( n:integer;Var v,a:RealArrayMpbyMp;
            Var nrot:integer;VAR d:RealArrayMp);

Label 99;
VAR
  iq,ip:integer;
  tresh,theta,tau,t,sm,s,h,g,c:Real;
  b,z:^RealArrayMp;

Begin
   new(b);
   new(z);
   FOR ip:= 1 to n do begin
      FOR iq:= 1 to n do v[ip,iq]:=0.0;
      v[ip,ip]:=1.0;
   End;
   For ip:= 1 to n DO BEGIN
   b^[ip]:=a[ip,ip];
   d[ip]:=b^[ip];
   z^[ip]:=0.0
   End;
   nrot:=0;
   FOr i:= 1 to 50 Do Begin
      sm:=0.0;
      For ip:= 1 to n-1 do
         For iq:=ip+1 to n Do
            sm:=sm+abs(a[ip,iq]);
      If sm =0.0 then GOTO 99;
      If i
      Else tresh :=0.0;
```

```
For ip:=1 to n-1 Do Begin
  For iq:=ip+1 to n Do Begin
    g:=100.0 *abs(a[ip,iq]);
    IF (i>4) and (abs(d[ip])+g = abs(d[ip]))
      and (abs(d[iq])+g = abs(d[iq])) then a[ip,iq] :=0
    Else IF abs(a[ip,iq])> tresh Then begin
       h:=d[iq]-d[ip];
       If abs(h)+g = abs(h) Then
         t:=a[ip,iq]/h
         Else begin
            theta:= 0.5*h/a[ip,iq];
                 t:=1.0/(abs(theta)+sqrt(1.0+sqr(theta)));
                 If theta < 0.0 then t:=-t
         End;
         c:=1.0/sqrt(1+sqr(t));
         s:=t*c;
         tau:=s/(1.0+c);
         h:=t*a[ip,iq];
         z^[ip]:=z^[ip]-h;
         z^[iq]:=z^[iq]+h;
         d[ip]:=d[ip]-h;
         d[iq]:=d[iq]+h;
         a[ip,iq]:=0.0;
         For j:=1 to ip-1 do Begin
            g:=a[j,ip];
            h:=a[j,iq];
            a[j,ip]:=g-s*(h+g*tau);
            a[j,iq]:=h+s*(g-h*tau)
         End;
         For j:=ip+1 to iq-1 Do begin
            g:=a[ip,j];
            h:=a[j,iq];
            a[ip,j]:=g-s*(h+g*tau);
            a[j,iq]:=h+s*(g-h*tau)
         End;
         For j:=iq+1 to n Do Begin
            g:=a[ip,j];
            h:=a[iq,j];
            a[ip,j]:=g-s*(h+g*tau);
            a[iq,j]:=h+s*(g-h*tau)
         End;
         For j:=1 to n Do Begin
            g:=v[j,ip];
            h:=v[j,iq];
            v[j,ip]:=g-s*(h+g*tau);
            v[j,iq]:=h+s*(g-h*tau)
         end;
         nrot:=nrot+1;
      end
   end
 End;
 For ip:=1 to n do Begin
```

```
            b^[ip]:=b^[ip]+z^[ip];
            d[ip]:=b^[ip];
            z^[ip]:=0.0
            End
         End;
         Writeln('pause in routine JACOBI');
         Writeln('50 iterations should not occur');
         readln;
      99:
        dispose(z);
        dispose(b);
      end;  {Procedure jaboci}


Procedure GetDataKeyboard(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
        var cvalue,Mu,xbar:mathmat.matx;
              var look:boolean;Var EigenValues,eigenvectors,SbyN,
              evalstar,evecstar:mathmat.matx;var ordp:integer;var rmatrix:mathmat.matx);

Begin
 If ordp=3 then
 Xdata.rows:=3
 else Xdata.rows:=2;

  cvalue.rows:=1; cvalue.cols:=1;
  cvalue.data[1,1]:=1.0;
 n.rows:=1; n.cols:=1;
  n.data[1,1]:=10.0;
GMatinput(n,2,0,'N',40,30,3,6,7);
 Xdata.cols:=Round(n.data[1,1]);

  For j:=1 to ordp Do Begin

     For k:=1 to Xdata.cols Do Begin
     XData.data[j,k]:=1.00;
     end;
     end;



GMatinput(cvalue,3,2,'c value',40,60,3,6,7);
If cvalue.data[1,1]
GMatinput(xdata,6,2,'x matrix',36,120,3,6,7);

Mat_Transpose(xdata,xdata_trans);



one.rows:=xdata.cols;one.cols:=1;
for j:=1 to one.rows Do Begin
 for k:=1 to one.cols do begin
 one.data[j,k]:=1.0;
```

```
  end;
  end;

Mat_Transpose(one,one_trans);

identity.rows:=xdata.cols;
identity.cols:=xdata.cols;

for j:=1 to xdata.cols do begin
  for k:= 1 to xdata.cols do begin
   identity.data[j,k]:=0.0;
   end; end;

for k:=1 to xdata.cols do begin
  identity.data[k,k]:=1.0;
  end;



MatMult(one,one_trans,onebyonet);
oneovern:=1.0/xdata.cols;
oneOverNMinusOne:=1.0/(xdata.cols-1.0);
mat_k_mult(onebyonet,temp,oneovern);
Matsub(identity,temp,temp2);

(* calculate X bar*)
mat_k_mult(one,temp3,oneovern);
matmult(xdata,temp3,xbar);

Mat_k_mult(xdata,temp,oneovernminusone);

Matmult(temp,temp2,temp3);

Matmult(temp3,xdata_trans,S);



for i:=1 to s.rows do begin
for j:=1 to s.cols do begin
   a[i,j]:=s.data[i,j];
    end;end;
(* Initialize D matrix *)
DMatrix.rows:=S.rows; DMatrix.cols:=S.Cols;
for j:=1 to DMatrix.rows Do Begin
For k:=1 to DMatrix.cols Do Begin
  DMatrix.data[j,k]:=0.0;
  end;end;
(* Calculate D matrix from S matrix *)
For j:=1 to DMatrix.rows Do begin
  DMatrix.data[j,j]:=sqrt(S.data[j,j]);
  end;
```

```
MatInvert(DMatrix,DInverse);

MatMult(DInverse,S,temp);
MatMult(temp,DInverse,RMatrix);
GMatwrite(RMatrix,5,4,'rho',50,285,3,6,7);

 Jac(ordp,v,a,nrot,d);

 EigenValues.rows:=ordp;Eigenvalues.Cols:=1;
 eigenvectors.rows:=ordp;Eigenvectors.Cols:=ordp;
 For j:=1 to ordp Do Begin
   (* allow neg ev *)
  Eigenvalues.data[j,1]:=d[j];
   end;
 FOr j:=1 to eigenvectors.rows do begin
   for k:=1 to eigenvectors.cols do begin
   eigenvectors.data[j,k]:=v[j,k];
   end; end;

(* This section uses the Eigen values and vectors to find *)
(* the endpoints of the major and minor axis of each ellipse *)
(* These are put into Xhigh and Xlow *)
(* endpoints of axis - each col is a coordinate for a point *)
(* 1st pt *)
xhigh[1,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xhigh[2,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 2nd Pt *)
xhigh[1,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xhigh[2,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
xlow[1,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xlow[2,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 5th Pt *)
xlow[1,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xlow[2,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

If ordp=3 then begin
xhigh[3,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xhigh[3,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 3rd pt *)
xhigh[1,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xhigh[2,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
```

```
xhigh[3,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];

xlow[3,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xlow[3,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 6th pt *)
xlow[1,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xlow[2,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xlow[3,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}

 (* Claculations for S/N  *)

   oneovern:=1.0/n.data[1,1];
Mat_k_mult(S,SbyN,oneovern);

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=SbyN.data[i,j];
   end;end;


jac(ordp,v,a,nrot,d);

EValStar.rows:=ordp;EValStar.Cols:=1;
eVecStar.rows:=ordp;EVecStar.Cols:=ordp;
For j:=1 to ordp Do Begin
 (* allow neg ev *)
 EValStar.data[j,1]:=d[j];
 end;
For j:=1 to evecStar.rows do begin
  for k:=1 to evecStar.cols do begin
  evecStar.data[j,k]:=v[j,k];
  end; end;
DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
For i:=1 to rmatrix.rows do begin
   For k:=1 to ordp do begin
   DsqrtEigenVals.data[i,k]:=0.0;
   end;end;
For i:=1 to ordp do begin
   DsqrtEigenVals.data[i,i]:=sqrt(abs(EValstar.data[i,1]));
   end;
Mat_transpose(evecstar,temp);

MatMult(DSqrtEigenVals,temp,temp2);
Matmult(evecstar,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
  for k:=1 to round(n.data[1,1])  do begin
  z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

  end;end;
```

```
matmult(Q,Z,temp);
MatAdd(temp,mu,xdata);
 (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xhigh[2,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 2nd Pt *)
 xhigh[1,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xhigh[2,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
 xlow[1,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xlow[2,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 5th Pt *)
 xlow[1,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xlow[2,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];



 If ordp=3 then begin
 xhigh[3,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xhigh[3,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
  (* 3rd pt *)
 xhigh[1,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xhigh[2,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];

 xhigh[3,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
 xlow[3,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xlow[3,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
  (* 6th pt *)
 xlow[1,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xlow[2,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
 xlow[3,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}


 (* OUTPUT *)
 GMatwrite(Eigenvalues,6,5,'Eigen Values',320,235,3,6,7);
 GmatWrite(Eigenvectors,6,5,'E vectors',320,285,3,6,7);
 Gmatwrite(xbar,5,4,'X Bar',320,190,3,6,7);
 Gmatwrite(S,5,4,'S Matrix',50,190,3,6,7);
 Gmatwrite(Sbyn,5,4,'S/n',40,235,3,6,7);
 OutTextXY(GetMaxX div 2-90,20,'PRESS ENTER FOR GRAPH');
 readln;clrscr;
 end; {GetDataKeyboard}
```

```
(* This Generates data for the theoretical case and mixed case *)

Procedure GenerateData(var txbar,xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
       var cvalue,Mu,xbar:mathmat.matx;
              var look:boolean;Var EigenValues,eigenvectors,SigmabyN,
              evalstar,evecstar:mathmat.matx;var ordp:integer;
              var sigma,sbyn,rmatrix:mathmat.matx);

Var

oneovern:real;

 Begin
randomize;

Mu.rows:=ordp; Mu.cols:=1;N.rows:=1;N.cols:=1;
xbar.rows:=ordp;xbar.cols:=1;txbar.rows:=ordp;txbar.cols:=1;
Cov.rows:=ordp; Cov.cols:=1;
Rmatrix.rows:=ordp; RMatrix.cols:=ordp;
N.data[1,1]:=10;
For i:=1 to Mu.rows do begin
   Mu.data[i,1]:=1;
   Cov.data[i,1]:=1;
   txbar.data[i,1]:=0;
   end;
For i:=1 to Cov.rows do begin
   For k:=1 to Rmatrix.Cols do begin
   Cov.data[i,k]:=1;
   RMatrix.data[i,k]:=1;
   End;end;
 CValue.rows:=1;CValue.Cols:=1;cvalue.data[1,1]:=1;

GMatinput(Mu,6,2,#230,46,69,3,6,7);
if look=true then
Gmatinput(Txbar,6,2,#230'.0',420,69,3,6,7);
GMatInput(Cov,6,5,#229'^2',260,240,3,6,7);
GMatInput(RMatrix,6,5,'Corr',36,120,3,6,7);
GMatInput(CValue,3,2,'C value',390,180,3,6,7);
If cvalue.data[1,1]
GmatInput(N,3,1,'N',260,180,3,6,7);

(* This section calculates the Sigma Matrix *)

sigma.rows:=ordp;sigma.cols:=ordp;
sigmabyn.rows:=ordp;sigmabyn.cols:=ordp;
s.rows:=ordp;s.cols:=ordp;
sbyn.rows:=ordp;sbyn.cols:=ordp;
For i:=1 to Rmatrix.rows do Begin
  For k:=1 to Rmatrix.cols do begin
  Sigma.data[i,k]:=Rmatrix.data[i,k]*sqrt(Cov.data[i,1])*sqrt(Cov.data[k,1]);
  end;end;
```

```
 For i:=1 to ordp do begin
    sigma.data[i,i]:=Cov.data[i,1];
    end;

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=sigma.data[i,j];
   end;end;


jac(ordp,v,a,nrot,d);

EigenValues.rows:=ordp;Eigenvalues.Cols:=1;
eigenvectors.rows:=ordp;Eigenvectors.Cols:=ordp;
For j:=1 to ordp Do Begin
  (* allow neg ev *)
 Eigenvalues.data[j,1]:=d[j];
  end;
For j:=1 to eigenvectors.rows do begin
   for k:=1 to eigenvectors.cols do begin
   eigenvectors.data[j,k]:=v[j,k];
   end; end;

DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
For i:=1 to rmatrix.rows do begin
    For k:=1 to ordp do begin
    DsqrtEigenVals.data[i,k]:=0.0;
    end;end;

For i:=1 to ordp do begin
    DsqrtEigenVals.data[i,i]:=sqrt(abs(Eigenvalues.data[i,1]));
    end;

Mat_transpose(eigenvectors,temp);

MatMult(DSqrtEigenVals,temp,temp2);
Matmult(eigenvectors,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
   for k:=1 to round(n.data[1,1])  do begin
   z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

   end;end;

xbar.rows:=ordp;xbar.cols:=round(n.data[1,1]);

for i:=1 to round(n.data[1,1]) do begin
   xbar.data[1,i]:=mu.data[1,1];
   xbar.data[2,i]:=mu.data[2,1];
   If ordp= 3 then
   xbar.data[3,i]:=Mu.data[3,1];
   end;
```

```
matmult(Q,Z,temp);
MatAdd(temp,xbar,xdata);
xbar.cols:=1;

 (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+Mu.data[1,1];
 xhigh[2,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+Mu.data[2,1];
 (* 2nd Pt *)
 xhigh[1,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+Mu.data[1,1];
 xhigh[2,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+Mu.data[2,1];

(* 4th pt *)
 xlow[1,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+Mu.data[1,1];
 xlow[2,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+Mu.data[2,1];
 (* 5th Pt *)
 xlow[1,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+Mu.data[1,1];
 xlow[2,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+Mu.data[2,1];



 If ordp=3 then begin
 xhigh[3,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+Mu.data[3,1];
 xhigh[3,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+Mu.data[3,1];
  (* 3rd pt *)
 xhigh[1,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+Mu.data[1,1];
 xhigh[2,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+Mu.data[2,1];
 xhigh[3,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+Mu.data[3,1];
 xlow[3,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+Mu.data[3,1];
 xlow[3,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+Mu.data[3,1];
 (* 6th pt *)
 xlow[1,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+Mu.data[1,1];
 xlow[2,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+Mu.data[2,1];
 xlow[3,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+Mu.data[3,1];


   end; {if}
 (* Claculations for Sigma/N *)

    oneovern:=1.0/n.data[1,1];
Mat_k_mult(Sigma,SigmabyN,oneovern);

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=SigmabyN.data[i,j];
    end;end;
```

```
jac(ordp,v,a,nrot,d);

EValStar.rows:=ordp;EValStar.Cols:=1;
eVecStar.rows:=ordp;EVecStar.Cols:=ordp;
For j:=1 to ordp Do Begin
 (* allow neg ev *)
 EValStar.data[j,1]:=d[j];
  end;
For j:=1 to evecStar.rows do begin
  for k:=1 to evecStar.cols do begin
  evecStar.data[j,k]:=v[j,k];
  end; end;
DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
For i:=1 to rmatrix.rows do begin
   For k:=1 to ordp do begin
   DsqrtEigenVals.data[i,k]:=0.0;
   end;end;
For i:=1 to ordp do begin
   DsqrtEigenVals.data[i,i]:=sqrt(abs(EValstar.data[i,1]));
   end;
Mat_transpose(evecstar,temp);

MatMult(DSqrtEigenVals,temp,temp2);
Matmult(evecstar,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
  for k:=1 to round(n.data[1,1]) do begin
  z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

  end;end;

(* This section uses the Eigen values and vectors to find *)
(* the endpoints of the major and minor axis of each ellipse *)
(* These are put into Xhigh and Xlow *)
(* endpoints of axis - each col is a coordinate for a point *)
(* 1st pt *)
xhigh[1,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+Mu.data[1,1];
xhigh[2,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+Mu.data[2,1];
(* 2nd Pt *)
xhigh[1,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+Mu.data[1,1];
xhigh[2,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+Mu.data[2,1];

(* 4th pt *)
xlow[1,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+Mu.data[1,1];
xlow[2,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+Mu.data[2,1];
(* 5th Pt *)
xlow[1,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+Mu.data[1,1];
xlow[2,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+Mu.data[2,1];


If ordp=3 then begin
```

```
xhigh[3,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+Mu.data[3,1];
xhigh[3,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+Mu.data[3,1];
 (* 3rd pt *)
xhigh[1,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+Mu.data[1,1];
xhigh[2,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+Mu.data[2,1];

xhigh[3,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+Mu.data[3,1];
xlow[3,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+Mu.data[3,1];
xlow[3,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+Mu.data[3,1];
(* 6th pt *)
xlow[1,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+Mu.data[1,1];
xlow[2,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+Mu.data[2,1];
xlow[3,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+Mu.data[3,1];
end; {if}

(* IF true then case #5 theory and empirical *)
 If look=true then begin

xdata.rows:=ordp;xdata.cols:=Round(n.data[1,1]);



   Mat_Transpose(xdata,xdata_trans);
one.rows:=xdata.cols;one.cols:=1;
for j:=1 to one.rows Do Begin
 for k:=1 to one.cols do begin
 one.data[j,k]:=1.0;
 end;
 end;

Mat_Transpose(one,one_trans);

identity.rows:=xdata.cols;

identity.cols:=xdata.cols;

for j:=1 to xdata.cols do begin
 for k:= 1 to xdata.cols do begin
  identity.data[j,k]:=0.0;
  end; end;

for k:=1 to xdata.cols do begin
 identity.data[k,k]:=1.0;
 end;



MatMult(one,one_trans,onebyonet);
oneovern:=1.0/xdata.cols;
oneOverNMinusOne:=1.0/(xdata.cols-1.0);
mat_k_mult(onebyonet,temp,oneovern);
Matsub(identity,temp,temp2);
```

```
(* calculate X bar*)
mat_k_mult(one,temp3,oneovern);
matmult(xdata,temp3,xbar);

Mat_k_mult(xdata,temp,oneovernminusone);

Matmult(temp,temp2,temp3);

Matmult(temp3,xdata_trans,S);


for i:=1 to s.rows do begin
for j:=1 to s.cols do begin
  a[i,j]:=s.data[i,j];
    end;end;
(* Initialize D matrix *)
DMatrix.rows:=S.rows; DMatrix.cols:=S.Cols;
for j:=1 to DMatrix.rows Do Begin
For k:=1 to DMatrix.cols Do Begin
  DMatrix.data[j,k]:=0.0;
  end;end;

(* Calculate D matrix from S matrix *)
For j:=1 to DMatrix.rows Do begin
  DMatrix.data[j,j]:=sqrt(S.data[j,j]);
  end;
MatInvert(DMatrix,DInverse);

MatMult(DInverse,S,temp);
MatMult(temp,DInverse,RMatrix);


 jac(ordp,v,a,nrot,d); (* get eigenvalues and Evectors*)

 Evalstar.rows:=ordp;Evalstar.Cols:=1;
 evecstar.rows:=ordp;Evecstar.Cols:=ordp;
 For j:=1 to ordp Do Begin
  (* allow neg ev *)
  Evalstar.data[j,1]:=d[j];
   end;
 FOr j:=1 to evecstar.rows do begin
   for k:=1 to evecstar.cols do begin
   evecstar.data[j,k]:=v[j,k];
   end; end;
  (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,7]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xhigh[2,7]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
```

```
(* 2nd Pt *)
xhigh[1,8]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xhigh[2,8]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
xlow[1,7]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xlow[2,7]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 5th Pt *)
xlow[1,8]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xlow[2,8]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];


If ordp=3 then begin
xhigh[3,7]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xhigh[3,8]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 3rd pt *)
xhigh[1,9]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xhigh[2,9]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,9]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
xlow[3,7]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xlow[3,8]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 6th pt *)
xlow[1,9]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xlow[2,9]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xlow[3,9]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}


  (* Claculations for S/N  *)

   oneovern:=1.0/n.data[1,1];
Mat_k_mult(S,SbyN,oneovern);

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=SbyN.data[i,j];
   end;end;


jac(ordp,v,a,nrot,d);

EValStar.rows:=ordp;EValStar.Cols:=1;
eVecStar.rows:=ordp;EVecStar.Cols:=ordp;
For j:=1 to ordp Do Begin
 (* allow neg ev *)
 EValStar.data[j,1]:=d[j];
 end;
For j:=1 to evecStar.rows do begin
  for k:=1 to evecStar.cols do begin
  evecStar.data[j,k]:=v[j,k];
```

```
   end; end;
 DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
 For i:=1 to rmatrix.rows do begin
    For k:=1 to ordp do begin
    DsqrtEigenVals.data[i,k]:=0.0;
    end;end;
 For i:=1 to ordp do begin
    DsqrtEigenVals.data[i,i]:=sqrt(abs(EValstar.data[i,1]));
    end;
 Mat_transpose(evecstar,temp);

 MatMult(DSqrtEigenVals,temp,temp2);
 Matmult(evecstar,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
   for k:=1 to round(n.data[1,1]) do begin
   z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

   end;end;

  (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,10]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xhigh[2,10]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 2nd Pt *)
 xhigh[1,11]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xhigh[2,11]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
 xlow[1,10]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xlow[2,10]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 5th Pt *)
 xlow[1,11]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xlow[2,11]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];


 If ordp=3 then begin
 xhigh[3,10]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xhigh[3,11]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 3rd pt *)
 xhigh[1,12]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xhigh[2,12]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,12]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
 xlow[3,10]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xlow[3,11]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 6th pt *)
 xlow[1,12]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xlow[2,12]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
```

```
xlow[3,12]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}

GMatWrite(S,5,4,'S',390,210,3,6,7);
GMatWrite(SbyN,5,4,'S/n',390,270,3,6,7);

end; (*if option 5*)
   (* OUTPUTS *)
GMATWRITE(xdata,6,2,'',20,30,3,6,7);
GMatwrite(Xbar,6,2,'Xbar',225,69,3,6,7);

GmatWrite(Sigma,5,3,#228,46,210,3,6,7);
GmatWrite(SigmabyN,5,3,#228'/N',46,270,3,6,7);
OutTextXY(GetMaxX div 2-90,20,'PRESS ENTER FOR GRAPH');
GMatwrite(RMatrix,5,4,'rho',290,120,3,6,7);

 readln;   clrscr;
 look:=false;
end;
 {GenerateData}



Procedure ReadDataEmpirical(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
        var cvalue,Mu,xbar:mathmat.matx;
               var look:boolean;Var EigenValues,eigenvectors,SbyN,
               evalstar,evecstar:mathmat.matx;var ordp:integer;var rmatrix:mathmat.matx);

Var
EmpData:text;

EmpData1:array[1..3,1..15] of real;
begin
 CValue.rows:=1;CValue.Cols:=1;cvalue.data[1,1]:=1;
 N.rows:=1;N.cols:=1;N.data[1,1]:=10;
GMatInput(CValue,3,2,'C Value',80,120,3,6,7);
If cvalue.data[1,1]

GmatInput(n,2,1,'N',GetMaxX DIV 2,120,5,4,7);

xdata.rows:=ordp;xdata.cols:=Round(n.data[1,1]);
Assign(EmpData,'c:\Empdata.dat');

   Reset(EmpData);


    for i:=1 to round(n.data[1,1])  do begin
    read(EmpData,EmpData1[1,i]); end;
     for i:=1 to round(n.data[1,1])  do begin
    read(EmpData,EmpData1[2,i]);
   end;
    If ordp=3 then begin
```

```
    for i:=1 to Round(n.data[1,1])  do begin
    read(EmpData,EmpData1[3,i]);
   end; {for}
   end;{if}
     for I:=1 to ordp do begin
     For J:=1 to round(n.data[1,1])  do begin
      xdata.data[I,J]:=EmpData1[I,J];
     End;end;


    GMATWRITE(xdata,6,2,'',20,30,3,6,7);

    Mat_Transpose(xdata,xdata_trans);
one.rows:=xdata.cols;one.cols:=1;
for j:=1 to one.rows Do Begin
  for k:=1 to one.cols do begin
  one.data[j,k]:=1.0;
  end;
  end;

Mat_Transpose(one,one_trans);

identity.rows:=xdata.cols;
identity.cols:=xdata.cols;

for j:=1 to xdata.cols do begin
  for k:= 1 to xdata.cols do begin
   identity.data[j,k]:=0.0;
   end; end;

for k:=1 to xdata.cols do begin
  identity.data[k,k]:=1.0;
  end;



MatMult(one,one_trans,onebyonet);
oneovern:=1.0/xdata.cols;
oneOverNMinusOne:=1.0/(xdata.cols-1.0);
mat_k_mult(onebyonet,temp,oneovern);
Matsub(identity,temp,temp2);

(* calculate X bar*)
mat_k_mult(one,temp3,oneovern);
matmult(xdata,temp3,xbar);

Mat_k_mult(xdata,temp,oneovernminusone);

Matmult(temp,temp2,temp3);

Matmult(temp3,xdata_trans,S);
```

```
for i:=1 to s.rows do begin
for j:=1 to s.cols do begin
  a[i,j]:=s.data[i,j];
    end;end;
(* Initialize D matrix *)
DMatrix.rows:=S.rows; DMatrix.cols:=S.Cols;
for j:=1 to DMatrix.rows Do Begin
For k:=1 to DMatrix.cols Do Begin
 DMatrix.data[j,k]:=0.0;
 end;end;

(* Calculate D matrix from S matrix *)
For j:=1 to DMatrix.rows Do begin
 DMatrix.data[j,j]:=sqrt(S.data[j,j]);
 end;
MatInvert(DMatrix,DInverse);

MatMult(DInverse,S,temp);
MatMult(temp,DInverse,RMatrix);
GMatwrite(RMatrix,5,4,'rho',50,285,3,6,7);


 jac(ordp,v,a,nrot,d);




 EigenValues.rows:=ordp;Eigenvalues.Cols:=1;
 eigenvectors.rows:=ordp;Eigenvectors.Cols:=ordp;
 For j:=1 to ordp Do Begin
  (* allow neg ev *)
  Eigenvalues.data[j,1]:=d[j];
  end;
 FOr j:=1 to eigenvectors.rows do begin
   for k:=1 to eigenvectors.cols do begin
   eigenvectors.data[j,k]:=v[j,k];
   end; end;
                              (*FIXED*)
 (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xhigh[2,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 2nd Pt *)
 xhigh[1,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xhigh[2,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
 xlow[1,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
```

```
xlow[2,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 5th Pt *)
xlow[1,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xlow[2,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];


If ordp=3 then begin
xhigh[3,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xhigh[3,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 3rd pt *)
xhigh[1,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xhigh[2,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
xlow[3,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xlow[3,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 6th pt *)
xlow[1,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xlow[2,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xlow[3,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}


  (* Claculations for S/N   *)

   oneovern:=1.0/n.data[1,1];
Mat_k_mult(S,SbyN,oneovern);

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=SbyN.data[i,j];
   end;end;


jac(ordp,v,a,nrot,d);

EValStar.rows:=ordp;EValStar.Cols:=1;
eVecStar.rows:=ordp;EVecStar.Cols:=ordp;
For j:=1 to ordp Do Begin
 (* allow neg ev *)
 EValStar.data[j,1]:=d[j];
 end;
For j:=1 to evecStar.rows do begin
  for k:=1 to evecStar.cols do begin
  evecStar.data[j,k]:=v[j,k];
  end; end;
DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
For i:=1 to rmatrix.rows do begin
  For k:=1 to ordp do begin
  DsqrtEigenVals.data[i,k]:=0.0;
```

```
    end;end;
 For i:=1 to ordp do begin
    DsqrtEigenVals.data[i,i]:=sqrt(abs(EValstar.data[i,1]));
    end;
 Mat_transpose(evecstar,temp);

 MatMult(DSqrtEigenVals,temp,temp2);
 Matmult(evecstar,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
   for k:=1 to round(n.data[1,1]) do begin
   z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

   end;end;



matmult(Q,Z,temp);
MatAdd(temp,mu,xdata);

  (* This section uses the Eigen values and vectors to find *)
 (* the endpoints of the major and minor axis of each ellipse *)
 (* These are put into Xhigh and Xlow *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xhigh[2,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 2nd Pt *)
 xhigh[1,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xhigh[2,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
 xlow[1,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
 xlow[2,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
 (* 5th Pt *)
 xlow[1,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
 xlow[2,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];



 If ordp=3 then begin
 xhigh[3,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xhigh[3,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 3rd pt *)
 xhigh[1,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xhigh[2,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
 xlow[3,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
 xlow[3,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
 (* 6th pt *)
 xlow[1,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
 xlow[2,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
```

```
xlow[3,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}


 close (Empdata);
 (* OUTPUT *)
 Gmatwrite(xbar,5,4,'X Bar',200,190,3,6,7);
Gmatwrite(S,5,4,'S Matrix',50,235,3,6,7);
OutTextXY(GetMaxX div 2-90,20,'PRESS ENTER FOR GRAPH');
GMatwrite(Eigenvalues,6,5,'Eigen Values',GetMaxX div 2,235,3,6,7);
 GmatWrite(Eigenvectors,6,5,'E vectors',GetMaxX div 2,285,3,6,7);
 readln;

end; {ReadData}


Procedure ReadDataTest(var xdata,S:Mathmat.matx;var xhigh,xlow:Real3by12array;
        var cvalue,Mu,xbar:mathmat.matx;
                var look:boolean;Var EigenValues,eigenvectors,SbyN,
                evalstar,evecstar:mathmat.matx;var ordp:integer;var rmatrix:mathmat.matx);

Var
sampOne:text;

Sample1:array[1..3,1..15] of real;
begin
 CValue.rows:=1;CValue.Cols:=1;cvalue.data[1,1]:=1;
 N.rows:=1;N.cols:=1;N.data[1,1]:=10;
 mu.rows:=ordp;Mu.cols:=1;
 For I:=1 to ordp do
 Mu.data[I,1]:=0.0;
GMatInput(CValue,3,2,'C Value',80,120,3,6,7);
If cvalue.data[1,1]
GmatInput(Mu,5,2,#230,80,190,3,6,7);
GmatInput(n,2,1,'N',GetMaxX DIV 2,120,5,4,7);

xdata.rows:=ordp;xdata.cols:=Round(n.data[1,1]);
Assign(sampOne,'c:\Samp1.dat');

    Reset(sampOne);


    for i:=1 to round(n.data[1,1])  do begin
    read(sampOne,sample1[1,i]); end;
```

```
  for i:=1 to round(n.data[1,1])  do begin
 read(sampOne,sample1[2,i]);
end;
If ordp=3 then begin
 for i:=1 to Round(n.data[1,1])  do begin
 read(sampOne,sample1[3,i]);
end;{for} end; {if}
     for I:=1 to ordp do begin
   For J:=1 to round(n.data[1,1]) do begin
    xdata.data[I,J]:=Sample1[I,J];
   End;end;


 GMATWRITE(xdata,6,2,'',20,30,3,6,7);

   Mat_Transpose(xdata,xdata_trans);
one.rows:=xdata.cols;one.cols:=1;
for j:=1 to one.rows Do Begin
 for k:=1 to one.cols do begin
 one.data[j,k]:=1.0;
 end;
 end;

Mat_Transpose(one,one_trans);

identity.rows:=xdata.cols;
identity.cols:=xdata.cols;

for j:=1 to xdata.cols do begin
 for k:= 1 to xdata.cols do begin
  identity.data[j,k]:=0.0;
  end; end;

for k:=1 to xdata.cols do begin
 identity.data[k,k]:=1.0;
 end;



MatMult(one,one_trans,onebyonet);
oneovern:=1.0/xdata.cols;
oneOverNMinusOne:=1.0/(xdata.cols-1.0);
mat_k_mult(onebyonet,temp,oneovern);
Matsub(identity,temp,temp2);

(* calculate X bar*)
mat_k_mult(one,temp3,oneovern);
matmult(xdata,temp3,xbar);

Mat_k_mult(xdata,temp,oneovernminusone);

Matmult(temp,temp2,temp3);
```

```
Matmult(temp3,xdata_trans,S);

Gmatwrite(xbar,5,4,'X Bar',260,190,3,6,7);

Gmatwrite(S,5,4,'S Matrix',50,235,3,6,7);

OutTextXY(GetMaxX div 2-90,20,'PRESS ENTER FOR GRAPH');
for i:=1 to s.rows do begin
for j:=1 to s.cols do begin
  a[i,j]:=s.data[i,j];
   end;end;
(* Initialize D matrix *)
DMatrix.rows:=S.rows; DMatrix.cols:=S.Cols;
for j:=1 to DMatrix.rows Do Begin
For k:=1 to DMatrix.cols Do Begin
 DMatrix.data[j,k]:=0.0;
 end;end;
(* Calculate D matrix from S matrix *)
For j:=1 to DMatrix.rows Do begin
 DMatrix.data[j,j]:=sqrt(S.data[j,j]);
 end;
MatInvert(DMatrix,DInverse);

MatMult(DInverse,S,temp);
MatMult(temp,DInverse,RMatrix);



 jac(ordp,v,a,nrot,d);




 EigenValues.rows:=ordp;Eigenvalues.Cols:=1;
 eigenvectors.rows:=ordp;Eigenvectors.Cols:=ordp;
 For j:=1 to ordp Do Begin
  (* allow neg ev *)
  Eigenvalues.data[j,1]:=d[j];
  end;
 FOr j:=1 to eigenvectors.rows do begin
   for k:=1 to eigenvectors.cols do begin
   eigenvectors.data[j,k]:=v[j,k];
   end; end;


 (*  This section uses the Eigen values and vectors to find *)
 (*  the endpoints of the major and minor axis of each ellipse *)
 (*  These are put into Xhigh and Xlow  *)
 (* endpoints of axis - each col is a coordinate for a point *)
 (* 1st pt *)
 xhigh[1,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
```

```
xhigh[2,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 2nd Pt *)
xhigh[1,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xhigh[2,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
xlow[1,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xlow[2,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 5th Pt *)
xlow[1,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xlow[2,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];



If ordp=3 then begin
xhigh[3,1]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xhigh[3,2]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 3rd pt *)
xhigh[1,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xhigh[2,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,3]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
xlow[3,1]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xlow[3,2]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 6th pt *)
xlow[1,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xlow[2,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xlow[3,3]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}

 (* Calculations for S/N  *)

   oneovern:=1.0/n.data[1,1];
Mat_k_mult(S,SbyN,oneovern);

for i:=1 to ordp do begin
for j:=1 to ordp do begin
  a[i,j]:=SbyN.data[i,j];
   end;end;


jac(ordp,v,a,nrot,d);

EValStar.rows:=ordp;EValStar.Cols:=1;
eVecStar.rows:=ordp;EVecStar.Cols:=ordp;
For j:=1 to ordp Do Begin
 (* allow neg ev *)
 EValStar.data[j,1]:=d[j];
  end;
For j:=1 to evecStar.rows do begin
  for k:=1 to evecStar.cols do begin
  evecStar.data[j,k]:=v[j,k];
  end; end;
```

```
DsqrtEigenVals.rows:=Rmatrix.rows;DsqrtEigenVals.cols:=Rmatrix.cols;
For i:=1 to rmatrix.rows do begin
  For k:=1 to ordp do begin
  DsqrtEigenVals.data[i,k]:=0.0;
  end;end;
For i:=1 to ordp do begin
  DsqrtEigenVals.data[i,i]:=sqrt(abs(EValstar.data[i,1]));
  end;
Mat_transpose(evecstar,temp);

MatMult(DSqrtEigenVals,temp,temp2);
Matmult(evecstar,temp2,Q);
Z.rows:=ordp;Z.cols:=round(n.data[1,1]);
For i:=1 to ordp do begin
  for k:=1 to round(n.data[1,1]) do begin
  z.data[i,k]:=sqrt(2.0*(ln(1/random)))*cos(2.0*pi*random);

  end;end;
 (* This section uses the Eigen values and vectors to find *)
(* the endpoints of the major and minor axis of each ellipse *)
(* These are put into Xhigh and Xlow  *)
(* endpoints of axis - each col is a coordinate for a point *)
(* 1st pt *)
xhigh[1,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xhigh[2,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 2nd Pt *)
xhigh[1,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xhigh[2,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];

(* 4th pt *)
xlow[1,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[1,1])+xbar.data[1,1];
xlow[2,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[2,1])+xbar.data[2,1];
(* 5th Pt *)
xlow[1,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[1,2])+xbar.data[1,1];
xlow[2,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[2,2])+xbar.data[2,1];


If ordp=3 then begin
xhigh[3,4]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xhigh[3,5]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 3rd pt *)
xhigh[1,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xhigh[2,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xhigh[3,6]:=cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
xlow[3,4]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[1,1]))*v[3,1])+xbar.data[3,1];
xlow[3,5]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[2,1]))*v[3,2])+xbar.data[3,1];
(* 6th pt *)
xlow[1,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[1,3])+xbar.data[1,1];
xlow[2,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[2,3])+xbar.data[2,1];
xlow[3,6]:=-cvalue.data[1,1]*(sqrt(abs(eigenvalues.data[3,1]))*v[3,3])+xbar.data[3,1];
end; {if}
```

```
 close (Sampone);
 GMatwrite(RMatrix,5,4,'rho',50,285,3,6,7);
 GMatwrite(Eigenvalues,6,5,'Eigen Values',GetMaxX div 2,235,3,6,7);
 GmatWrite(Eigenvectors,6,5,'E vectors',GetMaxX div 2,285,3,6,7);
 readln;clrscr;
end; {ReadData}


end.
```

Appendix G

The descriptive statistics are expressed and derived in matrix format. The three main statistics needed are:

1. Sample Means:

$$\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_p \end{bmatrix}$$

2. Sample variances and covariances:

$$s_n = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ s_{p1} & s_{p2} & \cdots & s_{22} \end{bmatrix}$$

3. Sample correlations:

$$r_n = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1p} \\ r_{21} & r_{22} & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ r_{p1} & r_{p2} & \cdots & r_{pp} \end{bmatrix}$$

Where p = the number of variables which corresponds to the number of dimensions to be displayed. The sample mean $\bar{x}$ is an unbiased estimate of the population mean. The sample variance covariance matrix s is the unbiased estimator for the population variance/covariance matrix Σ. Covariance and correlation measure the variable's linear associations.

First, the matrix definition of terms where gathered. The x matrix is the matrix of sample values for a $p = 3$ sample of n observations:

$$x_n = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ x_{31} & x_{32} & \cdots & x_{3n} \end{bmatrix}$$

The 1 matrix is a column matrix $p \times 1$ of 1's:

$$1_{3\times 1} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

I matrix:

$$I = \begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ 0\,0\,1 \end{bmatrix}$$

The *J* matrix is a $p \times p$ matrix of 1's:

$$\text{The identity matrix} = J_{3\times 3} \begin{bmatrix} 1\,1\,1 \\ 1\,1\,1 \\ 1\,1\,1 \end{bmatrix}$$

Eigenvalues are scalars $\lambda_1, \lambda_2 \ldots \lambda_p$

$$|A - \lambda I| = 0$$

$$|\chi| = \text{determinate of } \chi$$

EV_1 is a characteristic vector called eigenvector that is associated with a eigenvalue λ_1 such that :

$$Aev = \lambda ev$$

$$\begin{bmatrix} A_{11}\, A_{12} \\ A_{21}\, A_{22} \end{bmatrix} \begin{bmatrix} ev_1 \\ ev_2 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \begin{bmatrix} ev_1 \\ ev_2 \end{bmatrix}$$

At this point the idea of statistical distance needs to be discussed. Statistical distance is related to straight line of Euclidean distance. In Euclidean distance for $p = 2$ (2 dimensions), the distance between two points can be calculated using the Pythagorean theorem (see Figure G.1). The

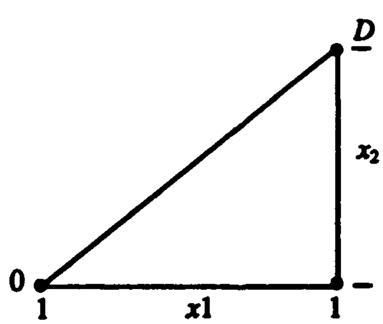


Figure G.1. Pythagorean Theorem

distance between point 0, which is assumed to be at the origin, and point *p* is $d(0, P)$.

$$d(0, P) = \sqrt{x_1^2 + x_2^2}$$

All points which lie a constant distance from *c*, from 0 satisfy (for $p = 2$).

The problem with using Euclidean distance measures to express statistical distances measures is that the Euclidean distance measure does not take into account the random sample's variability

involved with statistical study. The Euclidean measure assumes the coordinate of each point contributes equally to the calculation of distance. When dealing with statistics, it becomes necessary to weigh each coordinate according to the magnitude of its variability (as described in each s matrix) the correlation between its coordinates (as described in the r and R matrices). The statistical distance equations can be derived in the same manner as Euclidean distance. A random sample of points with greater variability in the x_1 direction then the v_2 direction, and $\bar{x}_1 = \bar{x}_2 = 0$ and x_1 independent from x_2 (i.e. $R_{12} = 0$) is shown in Figure G.2.

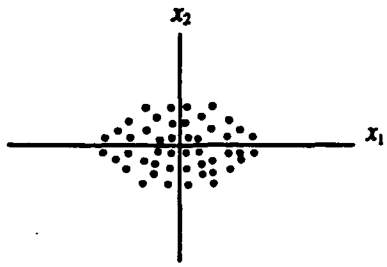


Figure G.2. Scatter Plot

Since there is more of a spread (higher variability in the x_1 direction) it is not "surprising" to find values of x_1 that are larger (in absolute terms) than values of x_2. Therefore to standardize the coordinates, each one (x_2 or v_2) is weighted differently. This is a form of standardization. The weights the standard deviates, therefore

$$x_1 = \frac{x_1}{\sqrt{s_{11}}} \text{ and } x_2 = \frac{x_2}{\sqrt{s_{22}}}$$

and

$$d(0, P) = \sqrt{x_1 + x_2} = \sqrt{\left(\frac{x_1}{\sqrt{s_{11}}}\right) + \left(\frac{x_2}{\sqrt{s_{22}}}\right)^2}$$

$$= \sqrt{\frac{x_1^2}{s_{11}} + \frac{x_2^2}{s_{22}}}$$

The points (x_1, x_2), which are an equal distance from point 0, fit the equation

$$\frac{x_1^2}{s_{11}} + \frac{x_2^2}{s_{22}} = c^2$$

This is an ellipse equation with its center at the origin (0,0), which is at the point $(\bar{x}_1\ \bar{x}_2)$. The major axis is along the x_1 axis. There is more variability in the x_1 axis. The major and minor axis are coincidental

with the x_1 and x_2 axis, respectfully. A more standard form of distance equation for the two points is given by:

$$d(P,Q) = \sqrt{\frac{(x_1-y_1)^2}{s_{11}} + \frac{(x_2-y_2)^2}{s_{22}} + \ldots + \frac{(x_p-y_p)^2}{s_{pp}}}$$

It must be remembered that this equation is based on the assumption of independent variables (i.e. $R_{12}=0$). In must relevant samples $R\,x_1\,x_p \neq 0$, therefore the effect of removing this assumption needs to be studied.

For a case of $p = 2$ and $R_{12} > 0$, the scatter plot in Figure 3.5 becomes the plot in Figure G.3.

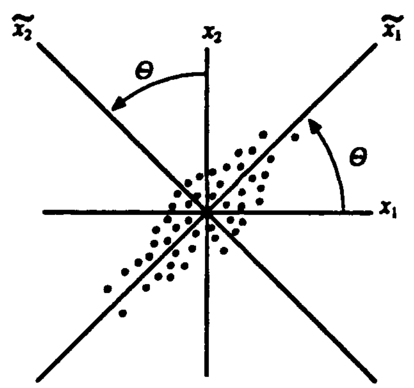


Figure G.3. Scatter plot with $R_{12} > 0$

The correlation $R_{12} = \dfrac{s_{12}}{\sqrt{s_{11}} \cdot \sqrt{s_{22}}} = \cos(\Theta_{12}) \cdot \Theta_{12} =$ the angle between $\tilde{x}_1$ and x_1 and the angle between $\tilde{x}_2$ and x_2

$$\Theta_{12} = \cos^{-1}(R_{12})$$

Therefore, the ellipse which describes the points equal distance (statistical distances) C from the center ($p = 2$) or ($p = 3$), has its major and minor axis rotate Θ_{ij} from the x_i, x_j axis when $R_{ij} \neq 0$ and the value of $\Theta_{ij} = \cos^{-1}(R_{ij})$.

$$d(0,P) = \sqrt{\frac{\tilde{x}_1^2}{\tilde{s}_{11}} + \frac{\tilde{x}_2^2}{\tilde{s}_{22}}}$$

where $\tilde{s}_{11}$ and $\tilde{s}_{22}$ are calculated variances with relationship to $\tilde{x}_1$ and $\tilde{x}_2$ respectively. Substituting these equations into the distances equation $d(0, P)$ yields

$$d(0,P) = \sqrt{a_{11}\,x_1^2\,2a_{12}\,x_1\,x_2 + a_{22}\,x_2^2}$$

where

$$a_{11} = \frac{\cos^2(\Theta)}{\cos^2(\Theta)\, s_{11} + 2\sin(\Theta)\cos(\Theta)\sin + \sin^2(\Theta)\, s_{22}}$$

$$+ \frac{\sin^2(\Theta)}{\cos^2(\Theta) - 2\sin(\Theta)\cos(\Theta)\, s_{12} + \sin^2(\Theta)\, s_{11}}$$

$$a_{22} = \frac{\sin^2(\Theta)}{\cos^2(\Theta)\, s_{11} + 2\sin(\Theta)\cos(\Theta)\, s_{12} + \sin^2(\Theta)\, s_{22}}$$

$$+ \frac{\cos^2(\Theta)}{\cos^2(\Theta)\, s_{22} - 2\sin(\Theta)\cos(\Theta)\, s_{12} + \sin^2(\Theta)\, s_{11}}$$

$$a_{12} = \frac{\cos^2(\Theta)\cos(\Theta)}{\cos^2(\Theta)\, s_{22} - 2\sin(\Theta)\cos(\Theta)\, s_{12} + \sin^2(\Theta)\, s_{22}}$$

$$- \frac{\sin(\Theta)\cos(\Theta)}{\cos^2(\Theta)\, s_{22} - 2\sin(\Theta)\cos(\Theta)\, s_{12} + \sin^2(\Theta)\, s_{11}}$$

The equation for an ellipse for $d(Q, P) = C$ and $R_{11} \neq 0$ becomes

$$c^2 = a_{11}(x_1 - y_2)^2 + 2a_{12}(x_1 - y_1)(x_2 - y_2) + a_{22}(x_2 - y_2)^2$$

In order to plot this ellipse the above equation must be stated as two relationships of x_2 in terms of x_1. This separation is important because the equation for the ellipse is not a function. For every value of x_1 y_1 pair there are two (x_2, y_2) pairs solves this equation. The equation is divided into a top and bottom half. Each representing one of two possible solutions. These relationships are:

Since the angle Θ is related to the correlation between the variables, it is important to show this concept to the student. The easiest way to show this is to display the major's and minor's axis end points. This can be done using the eigenvalues and eignvectors. Consider the ellipse in Figure G.4 The half length of the major axis is $c = \sqrt{\lambda_1}$ and the half length of the minor axis is $c = \sqrt{\lambda_2}$. When

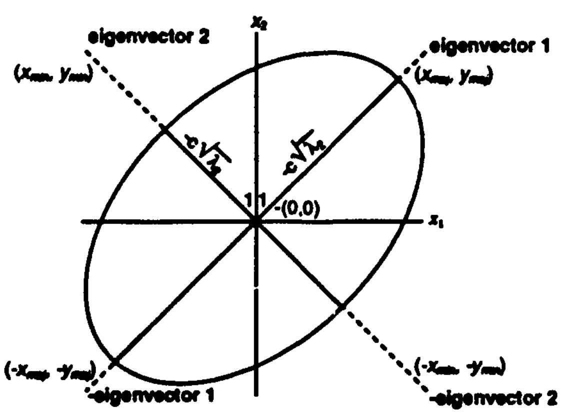


Figure G.4. End Points of Major and Minor Axis

you multiply these constants times the corresponding eignvector the result is the distance in the eigenvector's direction which yields the coordinates of the axis's. For example:

$$\text{eigenvector}_1 = \begin{bmatrix} x_1 \\ \overline{x_2} \end{bmatrix}$$

$$c \cdot \text{eigenvector}_1 + \sqrt{\lambda_1} \cdot = \begin{bmatrix} c\sqrt{\lambda_1}\, x_1 \\ c\sqrt{\lambda_2}\, x_2 \end{bmatrix}$$

$$= \text{coordinate of the major axis}$$

This is for a ellipse centered at the origin. If the ellipse is centered at $\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix}$ then the coordinate is given by

$$c\sqrt{\lambda_1} \cdot (\text{eigenvector}_1 = \bar{x}) = \begin{bmatrix} c\sqrt{\lambda_1}\,(x_1 + \bar{x}_1) \\ \sqrt{\lambda^1}\,(x_1 + \bar{x}_2) \end{bmatrix}$$

At this point all mathematical principles required to produce the program have been developed. The next phase consisted of developing the actual program.

Appendix H

This MathCad Template was used to develop and test the formulas and concepts needed for the development of the Pascal Program A3scrloo.exe. Plotting an Ellipse ORIGIN = 1

$$X1 := \begin{pmatrix} 26.7 & 38.4 & 19.2 & 20.6 & 18.9 & 14.8 & 19.0 & 14.2 & 13.7 & 7.7 \\ 3.3 & 2.4 & 1.7 & 1.0 & .9 & 1.0 & 2.7 & .8 & 1.1 & .2 \end{pmatrix}$$

$$X := \begin{pmatrix} 7 & 4 & 5 & 6.6 & 8.7 & 4.6 & 3.8 & 6.8 & 8.7 & 4.8 \\ 26.7 & 38.4 & 19.2 & 20.6 & 18.9 & 14.8 & 19.0 & 14.2 & 13.7 & 7.7 \\ 3.3 & 2.4 & 1.7 & 1.0 & .9 & 1.0 & 2.7 & .8 & 1.1 & .2 \end{pmatrix}$$

$n := \text{cols}(X)$ $\qquad$ $i := 1 .. \text{cols}(X)$

$one_i := 1$ $\qquad$ $xbar := X \cdot \dfrac{one}{n}$

$xone := \left(X^T\right)^{<1>}$ $\qquad$ $xtwo := \left(X^T\right)^{<2>}$ $\qquad$ $xthree := \left(X^T\right)^{<3>}$

$$xbar = \begin{pmatrix} 6 \\ 19.32 \\ 1.51 \end{pmatrix}$$

$$S := \frac{1}{n-1} \cdot X \cdot \left[\text{identity}(n) - \frac{1}{n} \cdot \left(one \cdot one^T\right) \right] \cdot X^T$$

$j := 1 .. \text{cols}(S)$

$$S = \begin{pmatrix} 3.313 & -3.258 & -0.46 \\ -3.258 & 70.411 & 5.873 \\ -0.46 & 5.873 & 0.97 \end{pmatrix}$$

$D12_{j,j} := \sqrt{S_{j,j}}$

$$D12 = \begin{pmatrix} 1.82 & 0 & 0 \\ 0 & 8.391 & 0 \\ 0 & 0 & 0.985 \end{pmatrix}$$

$D12INV := D12^{-1}$

$$D12INV = \begin{pmatrix} 0.549 & 0 & 0 \\ 0 & 0.119 & 0 \\ 0 & 0 & 1.015 \end{pmatrix}$$

$R := D12INV \cdot S \cdot D12INV$

$$R = \begin{pmatrix} 1 & -0.213 & -0.257 \\ -0.213 & 1 & 0.711 \\ -0.257 & 0.711 & 1 \end{pmatrix}$$

This angle - THETA12.angle is the angle for the relationship between and X2.

$THETA12 := acos(R_{2,1})$ $THETA12 = 1.786$ in Radians

$THETA12_{angle} := acos(R_{2,1}) \cdot \frac{180}{\pi}$ $THETA12_{angle} = 102.315$

This angle - THETA13.angle is the angle for the relationship between X1 and X3.

$THETA13 := acos(R_{3,1})$ $THETA13 = 1.83$ in Radians

$THETA13_{angle} := acos(R_{3,1}) \cdot \frac{180}{\pi}$ $THETA13_{angle} = 104.869$

This angle - THETA23.angle is the angle for the relationship between X2 and X3.

$THETA23 := acos(R_{2,3})$ $THETA23 = 0.78$ in Radians

$THETA23_{angle} := acos(R_{2,3}) \cdot \frac{180}{\pi}$ $THETA23_{angle} = 44.708$

$c := 2$

$s11 := S_{1,1}$ $s12 := S_{1,2}$ $s21 := S_{2,1}$ $s22 := S_{2,2}$

$s11 = 3.313$ $s12 = -3.258$ $s21 = -3.258$ $s22 = 70.411$

$s13 := S_{(1,3)}$ $s31 := S_{(3,1)}$ $s33 := S_{(3,3)}$ $s23 := S_{(2,3)}$

$s13 = -0.46$ $s31 = -0.46$ $s33 = 0.97$ $s23 = 5.873$

Calculate the EIGENVALUES AND EIGENVECTORS

$ev := \text{eigenvals}(S)$

$$ev = \begin{pmatrix} 3.167 \\ 71.063 \\ 0.464 \end{pmatrix}$$ These are the eigenvalues

$evec1 := \text{eigenvec}(S, ev_1)$

$evec2 := \text{eigenvec}(S, ev_2)$

$evec3 := \text{eigenvec}(S, ev_3)$

$$evec1 = \begin{pmatrix} -0.996 \\ -0.054 \\ 0.065 \end{pmatrix} \quad evec2 = \begin{pmatrix} -0.048 \\ 0.995 \\ 0.084 \end{pmatrix} \quad evec3 = \begin{pmatrix} 0.069 \\ -0.08 \\ 0.994 \end{pmatrix}$$

The above are the eigenvectors associated with the eigenvalues found above.

$c := 2$

$k1 := c \cdot \sqrt{ev_1}$ $k2 := c \cdot \sqrt{ev_2}$ $k3 := c \cdot \sqrt{ev_3}$

$k1 = 3.559$ $k2 = 16.86$ $k3 = 1.362$

$ax1 := k1 \cdot evec1 + xbar$ $a1xn := xbar - k1 \cdot evec1$

$ax2 := k2 \cdot evec2 + xbar$ $a2xn := xbar - k2 \cdot evec2$ $2 \cdot |k1 \cdot evec1| = 7.118$

$ax3 := k3 \cdot evec3 + xbar$ $a3xn := xbar - k3 \cdot evec3$ $2 \cdot |k2 \cdot evec2| = 33.72$

$ax1b := k2 \cdot evec1 + xbar$ $a1xnb := xbar - k2 \cdot evec1$ $2 \cdot |k3 \cdot evec3| = 2.724$

$ax2b := k1 \cdot evec2 + xbar$ $a2xnb := xbar - k1 \cdot evec2$

$ax3b := k2 \cdot evec3 + xbar$ $a3xnb := xbar - k2 \cdot evec3$

Below are the end points of the major and minor axis of the ellipsoid.

$$ax1 = \begin{pmatrix} 2.453 \\ 19.128 \\ 1.74 \end{pmatrix} \qquad a1xn = \begin{pmatrix} 9.547 \\ 19.512 \\ 1.28 \end{pmatrix}$$

$$ax2 = \begin{pmatrix} 5.184 \\ 36.101 \\ 2.921 \end{pmatrix} \qquad a2xn = \begin{pmatrix} 6.816 \\ 2.539 \\ 0.099 \end{pmatrix}$$

$$ax3 = \begin{pmatrix} 6.094 \\ 19.211 \\ 2.865 \end{pmatrix} \qquad a3xn = \begin{pmatrix} 5.906 \\ 19.429 \\ 0.155 \end{pmatrix}$$

$lengthmaj12 := c \cdot \sqrt{ev_1}$ $\quad lengthmax13 := c \cdot \sqrt{ev_1}$ $\quad lengthmax23 := c \cdot \sqrt{ev_2}$

$lengthmaj12 = 3.559$ $\quad lengthmax13 = 3.559$ $\quad lengthmax23 = 16.86$

$lengthmin12 := c \cdot \sqrt{ev_2}$ $\quad lengthmin13 := c \cdot \sqrt{ev_3}$ $\quad lengthmin23 := c \cdot \sqrt{ev_3}$

$lengthmin12 = 16.86$ $\quad lengthmin13 = 1.362$ $\quad lengthmin23 = 1.362$

You need to associate ev(1) with LARGEST SAMPLE VARIANCE max[s(ii)]

Compute the Coordinates of the ENDPOINTS of
the MAJOR and MINOR AXES of the Ellipse
and Plot the ELLIPSE using them

THIS CASE: s11 is largest Sample Variance so ev1 ~ s11
ev2 ~ s22

Compute maxv MATRIX {see my notes to you} ... and use
DIAGONAL VALUES of maxv to set min and max values for
range of INDEPENDENT VARIABLE of Plot (here x1)

$S_{1,1} = 3.313$ $\quad ev_1 = 3.167$ $\quad ev_3 = 0.464$

$S_{2,2} = 70.411$ $\quad ev_2 = 71.063$

$$maxvx1 := c\cdot\left(\sqrt{ev_1}\cdot evec1_1\right) + xbar_1 \qquad maxvx1 = 2.453$$

$$minvx1 := -c\cdot\left(\sqrt{ev_1}\cdot evec1_1\right) + xbar_1 \qquad minvx1 = 9.547$$

$$maxvx2 := c\cdot\left(\sqrt{ev_2}\cdot evec2_2\right) + xbar_2 \qquad maxvx2 = 36.101$$

$$minvx2 := -c\cdot\left(\sqrt{ev_2}\cdot evec2_2\right) + xbar_2 \qquad minvx2 = 2.539$$

Caution: We add XBAR to make points relative to CENTROID

Elliptical Functions Derived by MACSYMA...

This section does the calculations for X1 vs X2:

$$x2a := s12^2 - s11\cdot s22 \qquad x2b := 2\cdot s11\cdot s22 - 2\cdot s12^2$$

$$x2c := s12^2 - s11\cdot s22 \qquad xbar1 := xbar_1 \qquad xbar2 := xbar_2$$

$$x2a = -222.681 \qquad x2b = 445.362 \qquad x2c = -222.681$$

$$A1 := -s11\cdot xbar_2 \qquad A3(x1) := x2b\cdot x1\cdot xbar_1$$

$$A2 := x2a\cdot(xbar_1)^2 \qquad A3_{prime}(x1) := x2b\cdot xbar_1$$

$$A4(x1) := x2c\cdot x1^2 \qquad A5 := c^2\cdot s11^2\cdot s22 \qquad A6 := -c^2\cdot s11\cdot s12^2$$

$$A4_{prime}(x1) := 2\cdot x2c\cdot x1 \qquad A7(x1) := s12\cdot xbar_1 - s12\cdot x1$$

$$A7_{prime}(x1) := -s12$$

$$elipu12(x1) := -\frac{A1 + \sqrt{\left(A2 + x2b\cdot x1\cdot xbar_1 + x2c\cdot x1^2 + A5 + A6\right)} + s1\colon}{s11}$$

$$\text{elipu12}_{\text{prime}}(x1) := \frac{-\left[\left[\frac{1}{\left(2\cdot\sqrt{A2 + x2b\cdot x1\cdot xbar_1 + x2c\cdot x1^2 + A5 + A6}\right)}\cdot\left(\right.\right.\right.}{s11}$$

$$\text{prime}(x1) := \frac{d}{dx1}\,\text{elipu12}(x1)$$

$$\text{elipu12}_{\text{prime}}(9.64051) = 3640.49$$

$$\text{prime}(9.64051) = 3639.702$$

$$\text{elipb12}(x1) := \frac{-A1 + \sqrt{A2 + A3(x1) + A4(x1) + A5 + A6} - A7(x1)}{s11}$$

This section does the calculations for X1 vs X3:

$x3a := s13^2 - s11\cdot s33 \quad x3b := 2\cdot s11\cdot s33 - 2\cdot s13^2$

$x3c := s13^2 - s11\cdot s33 \quad xbar1 := xbar_1 \quad xbar3 := xbar_3$

$x2a = -222.681 \quad x2b = 445.362 \quad x2c = -222.681$

$B1 := -s11\cdot xbar_3 \;\; B2 := x3a\cdot(xbar_1)^2 \;\; B3(x1) := x3b\cdot x1\cdot xbar_1$

$B4(x1) := x3c\cdot x1^2 \quad B5 := c^2\cdot s11^2\cdot s33 \quad B6 := -c^2\cdot s11\cdot s13^2$

$$B7(x1) := s13\cdot xbar_1 - s13\cdot x1$$

$$\text{elipu13}(x1) := \frac{-\left(B1 + \sqrt{B2 + B3(x1) + B4(x1) + B5 + B6} + B7(x1)\right)}{s11}$$

$$\text{elipb13}(x1) := \frac{-B1 + \sqrt{B2 + B3(x1) + B4(x1) + B5 + B6} - B7(x1)}{s11}$$

This section does the calculations for X2 vs X3:

$x3a := s23^2 - s22 \cdot s33 \quad x3b := 2 \cdot s22 \cdot s33 - 2 \cdot s23^2$

$x3c := s23^2 - s22 \cdot s33 \quad xbar2 := xbar_2 \quad xbar3 := xbar_3$

$x3a = -33.797 \quad x3b = 67.594 \quad x3c = -33.797$

$C1 := -s22 \cdot xbar_3 \; C2 := x3a \cdot (xbar_2)^2 \quad C3(x2) := x3b \cdot x2 \cdot xbar_2$

$C4(x2) := x3c \cdot x2^{\wedge} C5 := c^2 \cdot s22^2 \cdot s33 \quad C6 := -c^2 \cdot s22 \cdot s23^2$

$C7(x2) := s23 \cdot xbar_2 - s23 \cdot x2$

$$\text{elipu23}(x2) := \frac{-\left(C1 + \sqrt{C2 + C3(x2) + C4(x2) + C5 + C6} + C7(x2)\right)}{s22}$$

$$\text{elipb23}(x2) := \frac{-C1 + \sqrt{C2 + C3(x2) + C4(x2) + C5 + C6} - C7(x2)}{s22}$$

$\delta := .1$

$low1 := minvx1 \quad xbar_1 = 6 \quad low2 := minvx2$

$xbar_2 = 19.32$

$high1 := maxvx1 \quad high2 := maxvx2$

$low1 = 9.547 \quad high1 = 2.453$

$high1 := if(maxvx1 > minvx1, maxvx1, minvx1)$

$low1 := if(maxvx1 > minvx1, minvx1, maxvx1)$

$high2 := if(maxvx2 > minvx2, maxvx2, minvx2)$

$low2 := if(maxvx2 > minvx2, minvx2, maxvx2)$

$low1 = 2.453$

$high1 = 9.547$ $\qquad THETA12_{angle} = 102.315$

$low1 = 2.453$ $\qquad lengthmaj12 = 3.559$

$hiah1 = 9.547$ $\qquad s := lengthmaj12 \cdot \cos(90 - THETA12_{angle})$

$high1 := 9.63$ $\qquad s = 3.447 \quad xbar_1 + s = 9.447$

$low1 := low1 - .01$

$\delta := .02$

$x1 := low1, low1 + \delta .. high1$

$prime(9.63) = 58.133$

$prime(xbar_1) = -0.983$

$prime(low1 - .0135) = -23.537$

$elipb12(6.09) = 35.623$

$elipu12(9.64051) = 15.72$

Note lengthmin>lengthmaj for 12

$$R = \begin{pmatrix} 1 & -0.213 & -0.257 \\ -0.213 & 1 & 0.711 \\ -0.257 & 0.711 & 1 \end{pmatrix} \qquad i := 1 .. 2$$

$endpointx1_1 := ax1_1$

$endpointx1_2 := a1xn_1$

$endpointx2_1 := ax1_2$

$endpointx2_2 := a1xn_2$

$endpointx21_2 := ax2_1$

$endpointx21_1 := a2xn_1$

$endpointx22_2 := ax2_2$

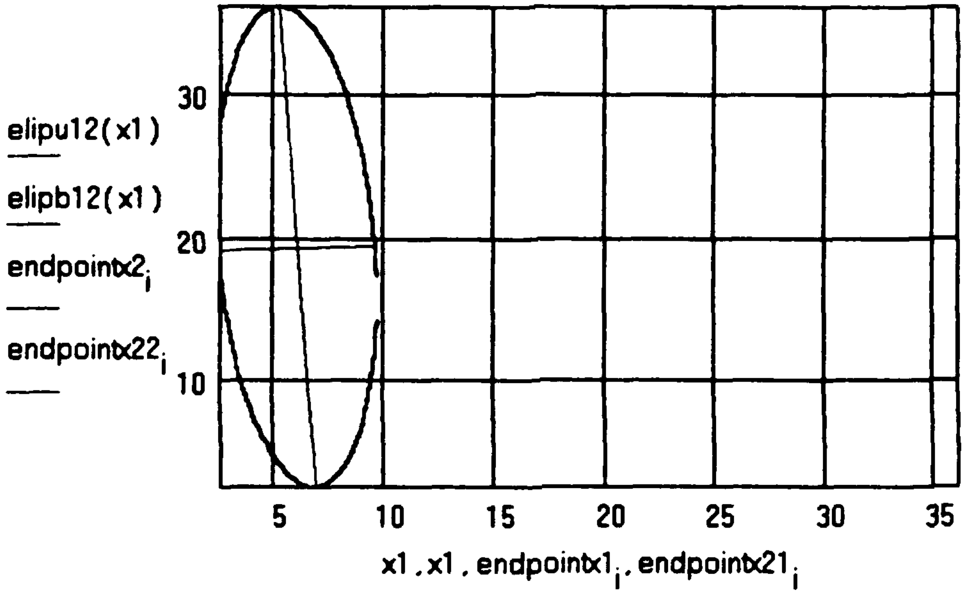
$endpointx22_1 := a2xn_2$

$$THETA12 - \frac{\pi}{2} = 12.315 \cdot deg$$

$$endpointx1 = \begin{pmatrix} 2.453 \\ 9.547 \end{pmatrix} \qquad endpointx21 = \begin{pmatrix} 6.816 \\ 5.184 \end{pmatrix}$$

$$endpointx2 = \begin{pmatrix} 19.128 \\ 19.512 \end{pmatrix} \qquad endpointx22 = \begin{pmatrix} 2.539 \\ 36.101 \end{pmatrix}$$

Graph of X1 vs X2



$\delta := .14$ $\qquad$ $i := 1..2$

$c := 2$ $\qquad$ $THETA13 = 104.869 \cdot deg$

$endpointx13_1 := ax1_1$ $\qquad$ $endpointx231_1 := ax3_1$

$endpointx13_2 := a1xn_1$ $\qquad$ $endpointx231_2 := a3xn_1$

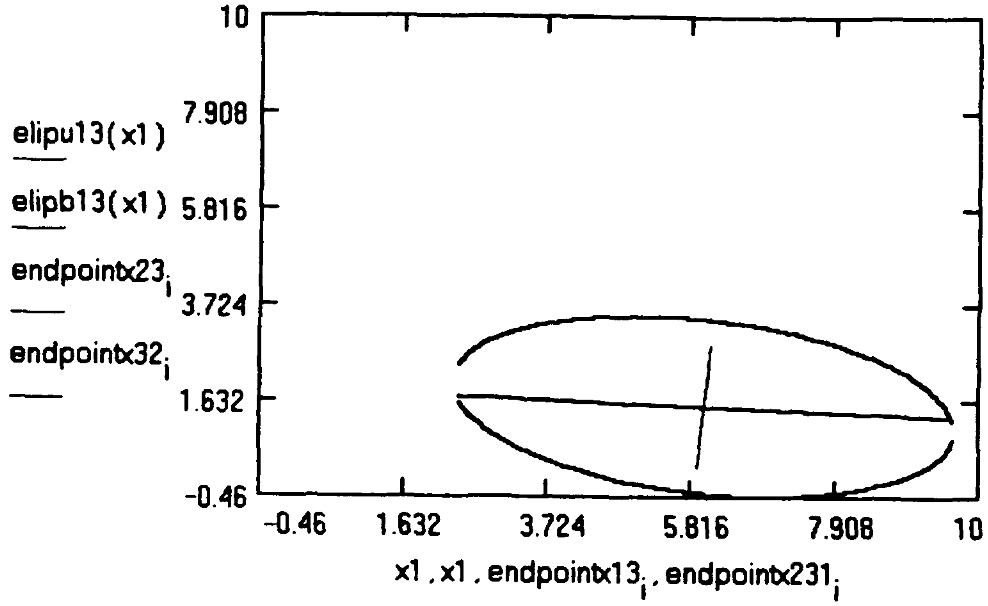
$endpointx23_1 := ax1_3$ $\qquad$ $endpointx32_1 := ax3_3$

$endpointx23_2 := a1xn_3$ $\qquad$ $endpointx32_2 := a3xn_3$

$$endpointx13 = \begin{pmatrix} 2.453 \\ 9.547 \end{pmatrix} \qquad endpointx231 = \begin{pmatrix} 6.094 \\ 5.906 \end{pmatrix}$$

$$endpointx23 = \begin{pmatrix} 1.74 \\ 1.28 \end{pmatrix} \qquad endpointx32 = \begin{pmatrix} 2.865 \\ 0.155 \end{pmatrix}$$

Graph of
X1 vs X3



$\delta := .14 \quad i := 1..2 \qquad xbar = \begin{pmatrix} 6 \\ 19.32 \\ 1.51 \end{pmatrix}$

$R_{(2,3)} = 0.711$

THETA23 = 44.708 · deg

lengthmin23 = 1.362

lengthmax23 = 16.86

$x2 := low2, low2 + \delta .. high2$

$endpointx2_1 := ax2_2 \qquad endptminx2_1 := ax3_2$

$endpointx2_2 := a2xn_2 \qquad endptminx2_2 := a3xn_2$

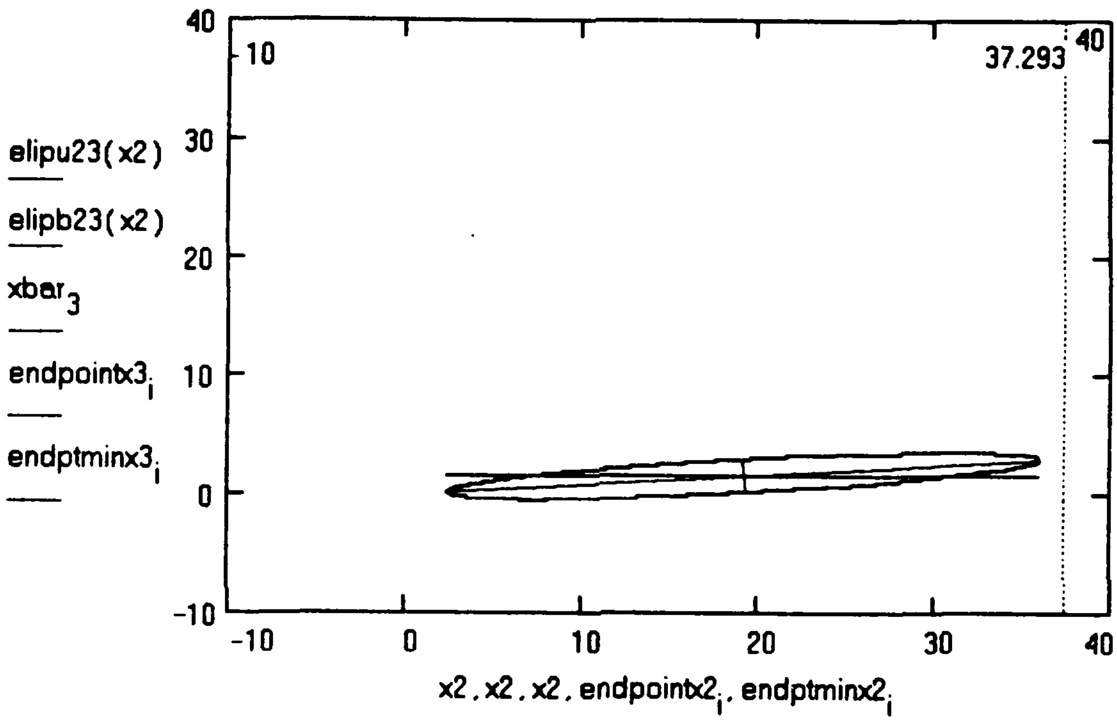
$endpointx3_1 := ax2_3 \qquad endptminx3_1 := ax3_3$

$endpointx3_2 := a2xn_3 \qquad endptminx3_2 := a3xn_3$

$endpointx2 = \begin{pmatrix} 36.101 \\ 2.539 \end{pmatrix} \qquad endptminx2 = \begin{pmatrix} 19.211 \\ 19.429 \end{pmatrix}$

$$\text{endpointx3} = \begin{pmatrix} 2.921 \\ 0.099 \end{pmatrix} \qquad \text{endptminx3} = \begin{pmatrix} 2.865 \\ 0.155 \end{pmatrix}$$

Graph of X2 vs X3



Bibliography

Ausubel, David P. *Educational Psychology: A Cognitive View.* New York: Holt, Rinehart and Winston, 1968.

Avermann, Donna E. and Cynthia R. Hynd. "Effects of Prior Knowledge Activation Modes and Text Structure on Nonscience Majors' Comprehension of Physics," *The Journal of Educational Research, 83(2)*: 97-101 (November/December 1989).

Bryant, Peter. "Geometry, Statistics, Probability: Variations on a Common Theme," *The American Statistician, 38(1)*: 38-48 (February 1984).

Cowley, Geoffrey and others. "Not Just for Nerds," *Newsweek*, 52-54 (April 9, 1990).

Halpern, Diane F. and others. "Analogies as an Aid to Understanding and Memory," *Journal of Educational Psychology*,82(2):298-305 (1990).

Hansard, Stone W. *An Interactive System of Computer Generated Graphic Displays for Motivating Meaningful Learning of Matrix Operations and Concepts of Matrix Algebra.* MS thesis, AFIT/GSM/ENC/90S-12. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1990 (AD-A229557).

Hirumi, Atsusi and Dennis R. Bowers. "Enhancing Motivation and Acquisition of Coordinate Concepts by Using Concept Trees," *The Journal of Educational Research, 84(5)*: 273-78 (May/June 1991).

Kardash, Carol Anne M. and others. "Effects of Schema on Both Encoding and Retrieval of Information From Prose," *Journal of Educational Psychology, 80(3)*: 324-328 (1988).

Kerr, Steven. "On the Folly of Rewarding A, While Hoping for B," *Academy of Management Journal, 18*: 769-83 (1975).

Margolis, Marvin S. "Perpendicular Projections and Elementary Statistics," *The American Statistician, 33(3)*: 131-135 (August 1979).

McCagg, Edward C. and Donald F. Dansereau. "A Convergent Paradigm for Examining Knowledge Mapping as a Learning Strategy," *The Journal of Educational Research, 84(6):* 317-24 (July/August 1991).

Novak, Joseph D. *A Theory of Education*. Ithica and London: Cornell University Press, 1986.

Novak, Joseph D. and D. Bob Gowin. *Learning How to Learn*. New York: Cambridge University Press, 1990.

Pearce, Stephen D. *An Application of Interactive Computer Graphics to the Study of Inferential Statistics and the General Linear Model.* MS Thesis, AFIT/GSM/ENC/91S-22. School of Systems and Logistics, Air Force Institute of Technology (AU),Wright-Patterson AFB, OH September 1991 (AD-A246668).

Press, William H. and others. *Numerical Recipes in Pascal, the Art of Scientific Computing.* New York: Cambridge University Press, 1989.

Saville, D.S. and G.R Wood. "A Method for Teaching Statistics Using N-Dimensional Geometry," *The American Statistician, 40(3):* 205-213 (August 1986).

Scherkenbach, William W. *The Deming Route to Quality and Productivity.* Washington DC: CEEPress Books, George Washington University, 1988.

Schmid, Richard F. and Giovanni Telaro. "Concept Mapping as an Instructional Strategy for High School Biology," *The Journal of Educational Research, 84(2)*: 78-84, (November/December 1990).

Schonberger, Richard J. *World Class Manufacturing, The Lessons of Simplicity Applied.* New York: The Free Press, a division of Macmillan, Inc., 1986.

Schwab, J. "The Practical 3: Transition into Curriculum," *School Review, 81(4)*: 501-22, (1973).

Vosniadou, Stella and Marlene Schommer. "Explanatory Analogies Can Help Children Acquire Information From Expository Text," *Journal of Educational Psychology, 80(4)*: 524-535 (1988).

Vita

Captain Ronald G. Garlicki was born on 6 December 1962 in Pittsburgh, Pennsylvania. He graduated from North Hills High School in Pittsburgh in 1981. He attended The Pennsylvania State University where he received a two year Air Force Reserve Officer Training Corps Scholarship. In 1985 he graduated with a Bachelor of Science degree in Electrical Engineering. His first assignment was to the F-16 System Program Office (SPO) at Wright-Patterson AFB, Ohio as an Integration Manager. Later he was assigned to the Ground Launched Cruise Missile SPO and then the Advanced Tactical Fighter SPO as an Electronic Combat Program manager. In November 1989 he married Diana Sue Travelute. In May 1991 he entered the Air Force Institute of Technology at Wright-Patterson AFB, Ohio, where he worked to obtain a Masters of Science in Systems Management degree.

5573 Oak Valley Road
Kettering, OH 45440-2332

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

[illegible]

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1992	Master's Thesis

4. TITLE AND SUBTITLE

INTERACTIVE GRAPHICS SYSTEM FOR THE STUDY OF VARIANCE/COVARIANCE STRUCTURES OF BIVARIATE AND MULTIVARIATE NORMAL POPULATIONS

5. FUNDING NUMBERS

6. AUTHOR(S)

Ronald G. Garlicki, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GSM/ENC/92S-9

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This research created a graphics oriented computer program which was used as part of a Visualization, Verbalization, Algorithization and Mathematization (VVAM) learning protocol. The curriculum for this research was the study of variance/covariance structures of bivariate and multivariate normal populations. The program displays the geometric images corresponding to the various possible covariance structures. These images facilitate and encourage experiment-based self discovery learning. The program encourages the student to take an active role in their own education. The program created is self contained, calculating all the statistical values it requires to create the various geometric images. The student has complete control in choosing what scenario they wish to study. The program was tested using four specific scenarios which represented a cross section of all possible scenarios. Within each of these scenarios were several options which were designed to encapsulate different aspects of the curriculum. The results of the research showed that the program, under the aegis of the VVAM, does facilitate the visualization and verbalization of the complex mathematical concepts associated with covariance structures. The learning environment was found to promote the creation of new knowledge on three distinct levels.

14. SUBJECT TERMS

Statistics, Learning, Computer, Covariance, Variance, Graphics, Geometric Approach

15. NUMBER OF PAGES 205

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassifed	Unclassifed	Unclassifed	UL

NSN 7540-01-280-5500

Standard Form 298

AFIT Control Number AFIT/GSM/ENC/92S-9

AFIT RESEARCH ASSESSMENT

The purpose of this questionnaire is to determine the potential for current and future applications of AFIT thesis research. Please return completed questionnaires to: AFIT/LSC, Wright-Patterson AFB OH 45433-9905.

1. Did this research contribute to a current research project?

a. Yes b. No

2. Do you believe this research topic is significant enough that it would have been researched (or contracted) by your organization or another agency if AFIT had not researched it?

a. Yes b. No

3. The benefits of AFIT research can often be expressed by the equivalent value that your agency received by virtue of AFIT performing the research. Please estimate what this research would have cost in terms of manpower and/or dollars if it had been accomplished under contract or if it had been done in-house.

Man Years ______________ $ ______________

4. Often it is not possible to attach equivalent dollar values to research, although the results of the research may, in fact, be important. Whether or not you were able to establish an equivalent value for this research (3, above) what is your estimate of its significance?

a. Highly Significant b. Significant c. Slightly Significant d. Of No Significance

5. Comments

______________________________ ______________________________
Name and Grade Organization

______________________________ ______________________________
Position or Title Address

DEPARTMENT OF THE AIR FORCE
AFIT/LSC Bldg 642
2950 P St
45433-7765

OFFICIAL BUSINESS

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 1006 DAYTON OH

POSTAGE WILL BE PAID BY U.S. ADDRESSEE

Wright-Patterson Air Force Base

AFIT/LSC Bldg 642
2950 P St
Wright Patterson AFB OH 45433-9905